

NASA Contractor Report 178339

HALOE TEST AND EVALUATION SOFTWARE

W. Edmonds  
S. Natarajan

ST Systems Corporation (STX)  
28 Research Drive  
Hampton, VA 23666

Contract NAS1-18022

June 1987

(NASA-CR-178339) HALOE TEST AND EVALUATION  
SOFTWARE (ST Systems Corp.) 142 p Avail:  
MIS BC A07/MF A01 CSCI 09B

N87-26575

Unclas  
G3/61 0090709



National Aeronautics and  
Space Administration

Langley Research Center  
Hampton, Virginia 23665-5225

## TABLE OF CONTENTS

	<u>Page</u>
Abstract.....	iii
SECTION 1 - INTRODUCTION.....	1-1
SECTION 2 - HALOE.....	2-1
SECTION 3 - SOFTWARE DEVELOPMENT.....	3-1
SECTION 4 - DATA ACQUISITION.....	4-1
SECTION 5 - SYSTEM DESIGN.....	5-1
SECTION 6 - DATA REDUCTION & ANALYSIS.....	6-1
APPENDIX A - HARP.....	A-1
APPENDIX B - HPLOT.....	B-1
APPENDIX C - SPECRES.....	C-1
APPENDIX D - SPECPLT.....	D-1
APPENDIX E - MONITOR.....	E-1

## Abstract

Computer programming, system development and analysis efforts during this contract were carried out in support of the Halogen Occultation Experiment (HALOE) at NASA/Langley. Support in the major areas of data acquisition and monitoring, data reduction and system development are described along with a brief explanation of the HALOE project. Documented listings of major software are located in the appendix.

PRECEDING PAGE BLANK NOT FILMED

## SECTION 1 - INTRODUCTION

Support of the Halogen Occultation Experiment (HALOE) during this level-of-effort contract consisted of computer programming, system design, data acquisition, data reduction and data analysis efforts.

HALOE is briefly described in Section 2 of this final report. Section 3 covers computer programming developments. Section 4 describes data acquisition support. System design is reviewed in Section 5, and Section 6 covers data reduction and data analysis support. Listings of programs are in the appendix.

## SECTION 2 - HALOE

The objective of the Halogen Occultation Experiment is to measure trace constituents of the upper atmosphere to determine the mechanism of ozone depletion. The HALOE instrument was designed to measure these gases using a solar occultation technique. Utilizing four gas correlation and four bolometer channels, the HALOE instrument will view the sun during orbital sunrise and sunset events to measure the spectral occultation caused by ozone, water vapor, nitrogen dioxide, carbon dioxide, hydrogen fluoride, hydrogen chloride, methane and nitric oxide. Knowledge of the distribution of these gases on a global level over a long period of time should provide the means to better understand the mechanism of ozone depletion. HALOE will be one of Ten instruments on UARS (Upper Atmosphere Research Satellite) currently scheduled for launch aboard the space shuttle from KSC in 1991.

### SECTION 3 - SOFTWARE DEVELOPMENT

A number of computer programs were developed under this contract to support the testing and characterization of the HALOE instrument. A variety of computer systems and languages were used to accomplish these tasks. Computer hardware included HP-1000, IBM-XT and CDC Cyber computers. Computer languages utilized were FORTRAN, PASCAL, FORTH and IBM assembler.

The HALOE black body life test was supported with the development of a program called "HPLOT" on the CDC NOS facility. "HPLOT" (written in FORTRAN 5) plots the various black body parameters against the PRT (platinum resistance thermometer) and tabulates daily averages of all the parameters (see appendix for program listing and sample output).

"HARP" was developed on the HP1000 in FORTRAN to aid in the analysis of HALOE test data tapes. HARP will process data directly from tape or from disc files previously derived from test tapes. Data windowing features allow the user to select time segments for processing and/or archival to disc. Annotate records can be searched in a forward or reverse direction to locate significant events for processing. Plot files containing selected parameters can be created for another program "UPLOT" to plot on the HP pen plotter, or on the CRT. A statistics option allows the user to select parameters for statistical analysis and tabulation.

Using Turbo Pascal on an IBM-XT fitted with a Lab Master card, software was developed to acquire data from the HALOE GCETS

(Gas Correlation Electronic Test Set).

Several versions of this software were created to acquire data for IFOV, balance-linearity, spectral response and NO noise tests. Data acquired by these programs was written to disc files. Plotter programs were developed to generate plots of the data on an HP pen plotter connected to an IEEE-488 card in the IBM-XT. LaRCNET was used to transfer some of these data files to NOS for analysis by the HALOE science team.

During this contract, work was begun on software which will monitor the HALOE data stream on a real time basis. Data will be transferred from the HP1000 to the IBM-XT over an IEEE-488 bus (HPIB) and displayed on a color monitor in color coded form. Red or yellow will indicate out-of-limit conditions, while green or white will indicate acceptable values. The computer language "FORTH" was used to develop the communications between the HP1000 and the IBM, and Turbo Pascal was used to write the display software for the IBM. Listings and sample output from some of the significant pieces of software are contained in the appendix to this report.

#### SECTION 4 - DATA ACQUISITION

Data acquisition support activities were performed under this contract for the following specialized tests of the HALOE instrument: IFOV, balance-linearity, spectral response and NO noise testing.

For the IFOV tests, measurements were made in azimuth and elevation for the gas correlation channels: HCl, HF, CH<sub>4</sub>, NO (both gas and vacuum) and for the bolometer channels: H<sub>2</sub>O, CO<sub>2</sub>, NO<sub>2</sub>, O<sub>3</sub>. Results were tabulated and plotted immediately following each elevation or azimuth test (see sample plot).

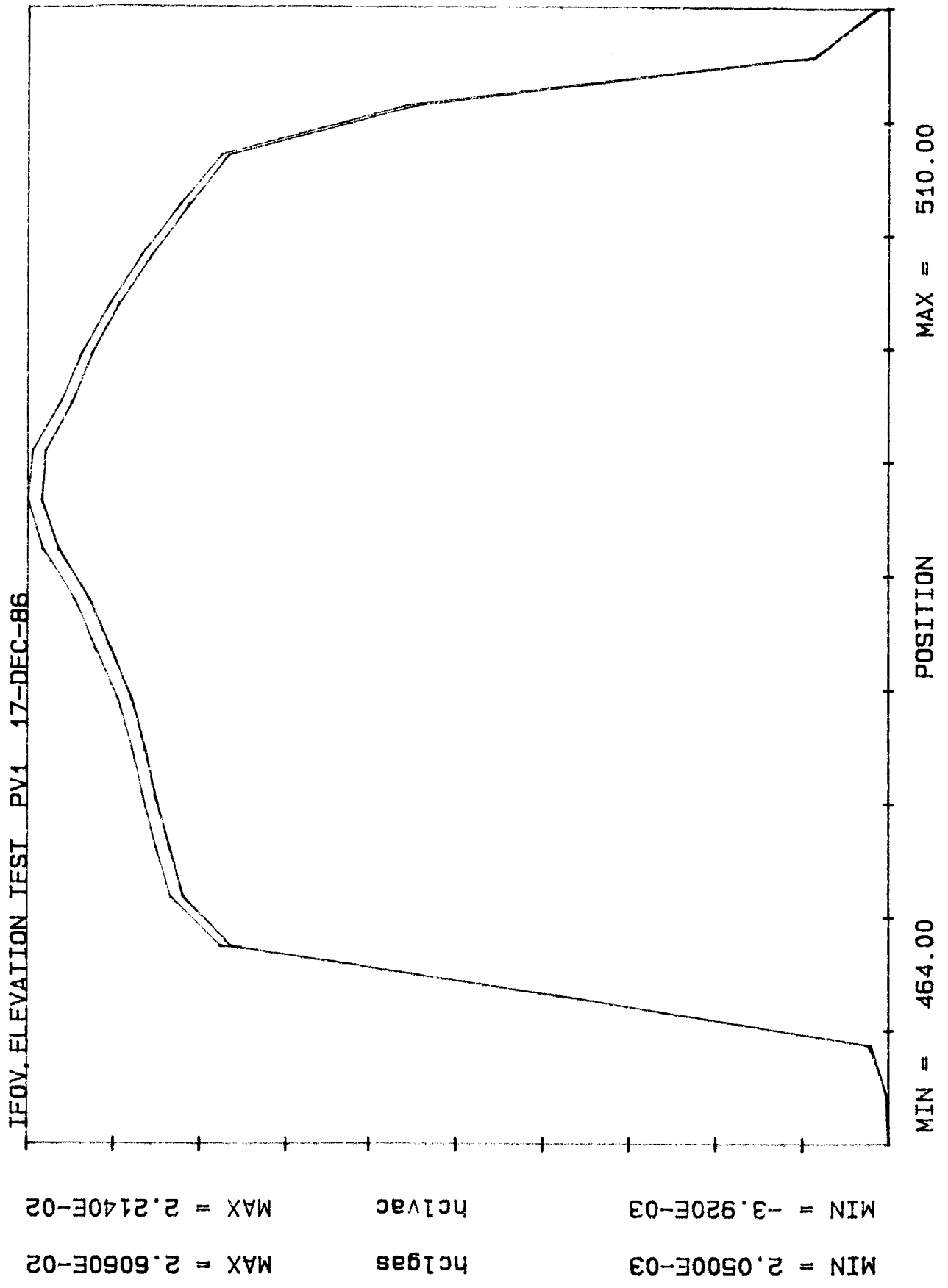
Balance-linearity test data were acquired in a similar manner. To determine the linearity of each channel, correlation coefficients were calculated and printed out immediately following each test. Test data were also sent to the CDC NOS facility for further evaluation. Data was acquired for these tests using software developed under this contract (described elsewhere in this document) on an IBM-XT fitted with a Tecmar Lab Master data acquisition card.

NO noise testing was accomplished by monitoring the NO channels (vac. & gas) during a series of manipulations of the instrument and associated equipment in the clean room.

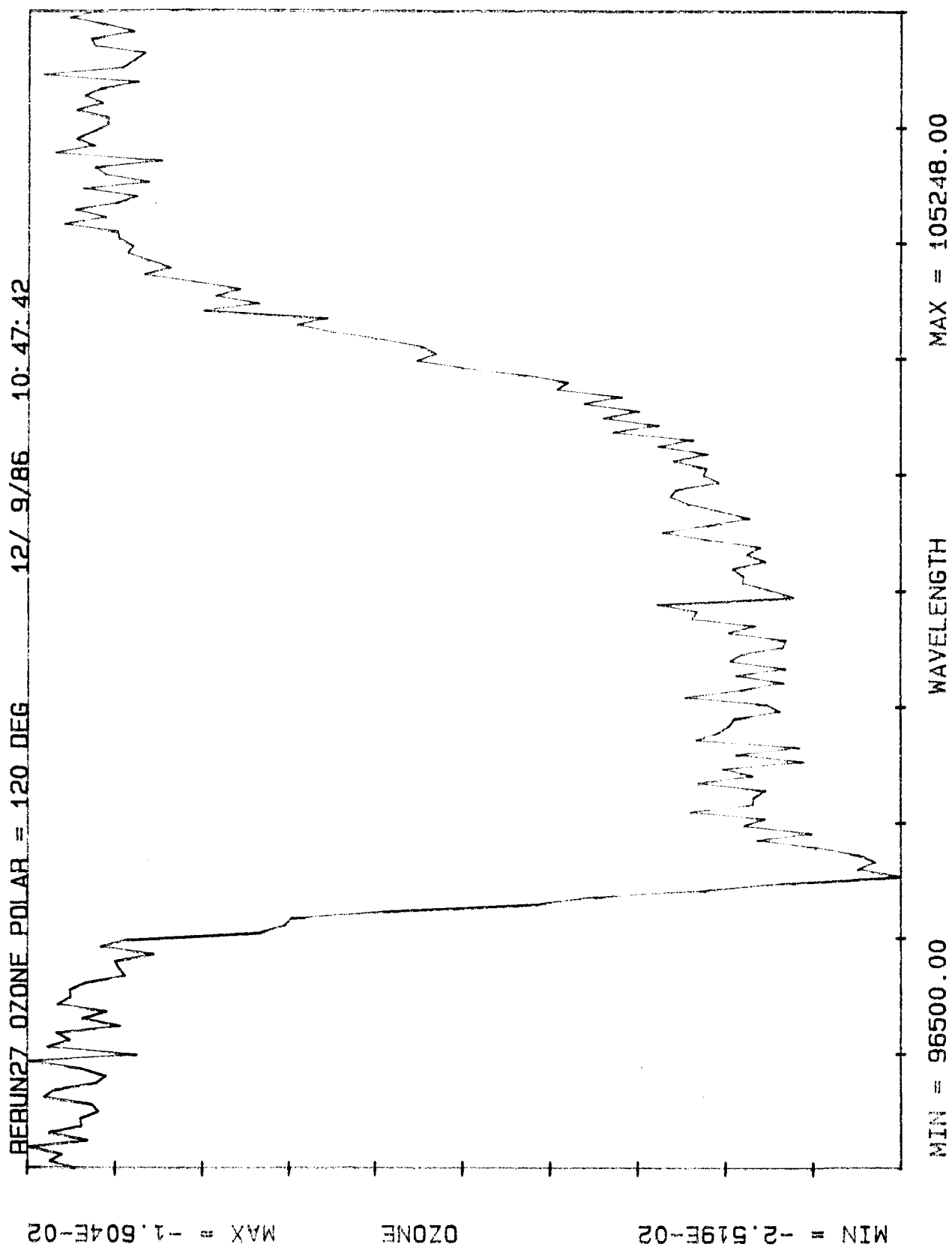
Data acquisition efforts for the spectral response tests involved the use of additional software and hardware. In addition to the Lab Master software and hardware for data acquisition from the GCETS, the IBM-XT needed to communicate with the CD2A compudrive. This RS232 communications allowed the IBM-

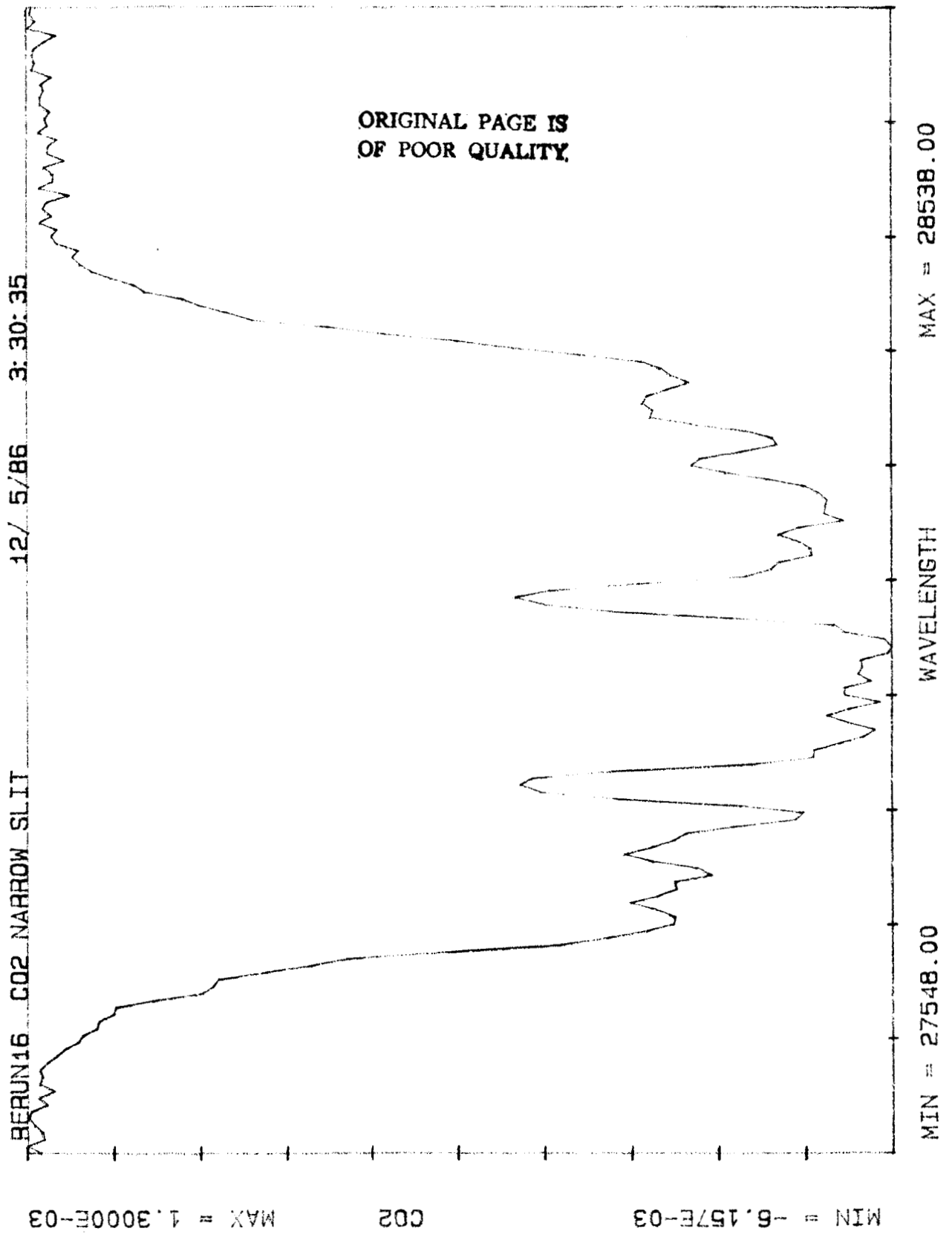


XT to detect when the spectrometer changed wavelength. Each step in wavelength was then used to trigger the acquisition of data from the GCETS. Data, including the wavelength, was then saved to disc for immediate processing after each spectral test. Plots were generated with the IBM and an HP pen plotter. The data was also sent to ACD using LaRCnet for further study by the science team (see sample spectral response plot and the data acquisition block diagram which follow).

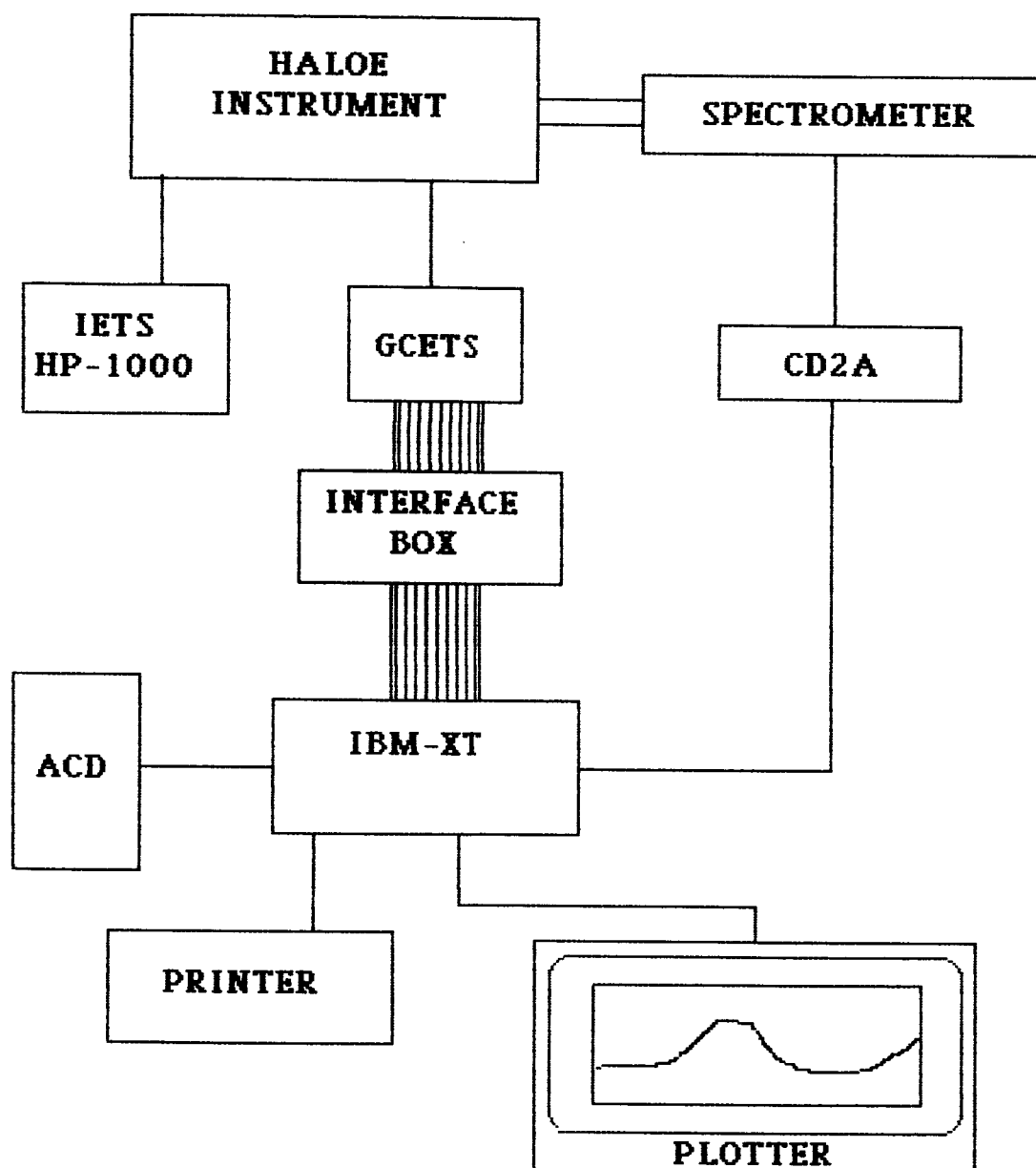


ORIGINAL PAGE IS  
OF POOR QUALITY

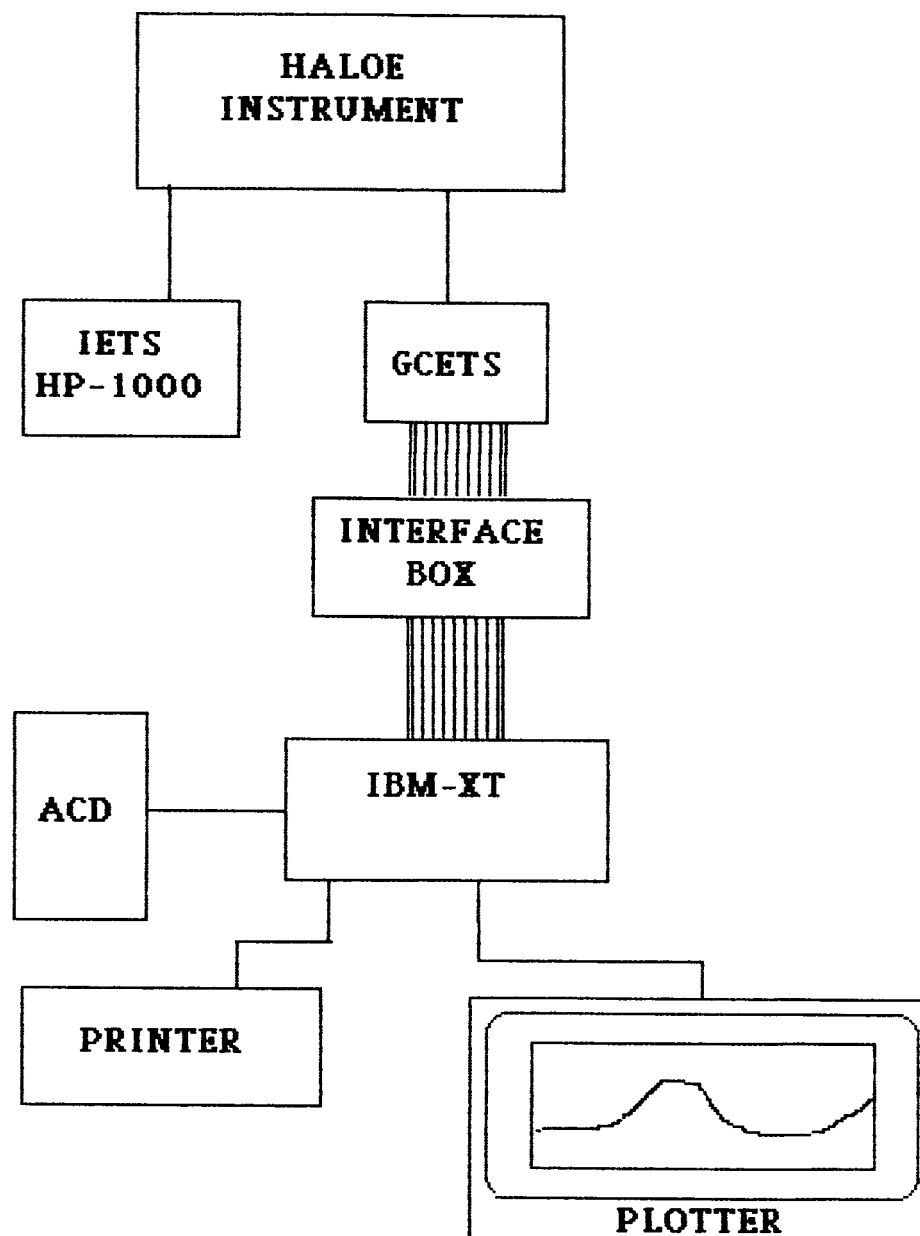




# HALOE SPECTRAL RESPONSE DATA ACQUISITION SET-UP



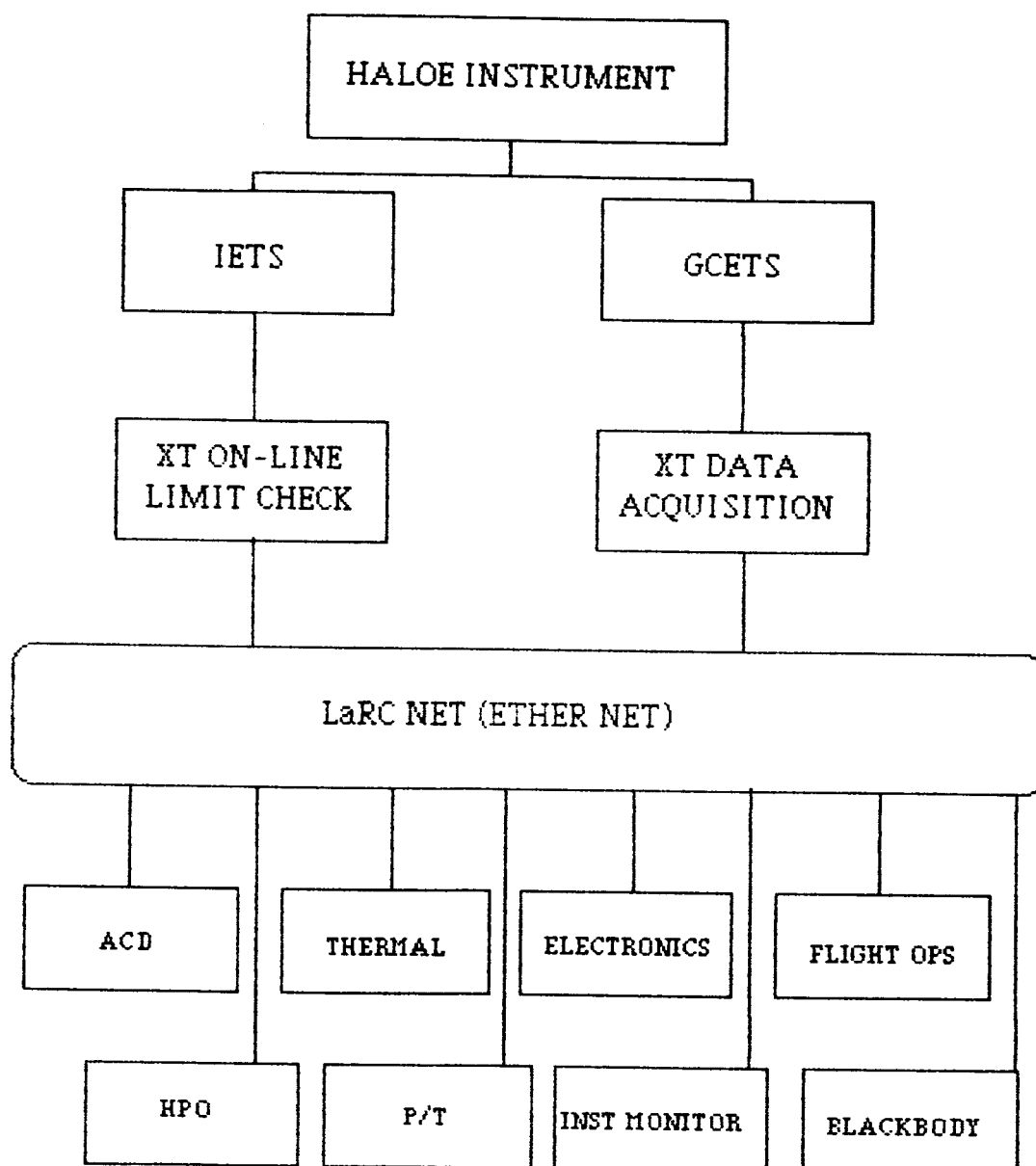
# HALOE IFOV & BALANCE-LINEARITY DATA ACQUISITION SET-UP



## SECTION 5 - SYSTEM DESIGN

Considerable effort was made during this contract to design and implement a system for quick-look data reduction during the remaining testing at Langley and during satellite integration and testing when HALOE is installed on UARS (Upper Atmosphere Research Satellite). The attached block diagrams show the hardware configuration which was proposed and which will be assembled, tested and utilized under a subsequent contract. Some of the software requirements for this system were partially completed during this contract and will be finished early in the new contract period. Other system development work was done in the evaluation of an automated test control system. Although insufficient time and resources were available to fully design and implement such a system, a useful subset was designed and implemented on the HP1000 IETS. This system involved the use of FORTH (a computer language). FORTH facilitated the construction of commands and combinations of commands which could be issued to the HALOE instrument during tests. (These efforts were done under a separate STX contract and were accomplished by Milton Fabert).

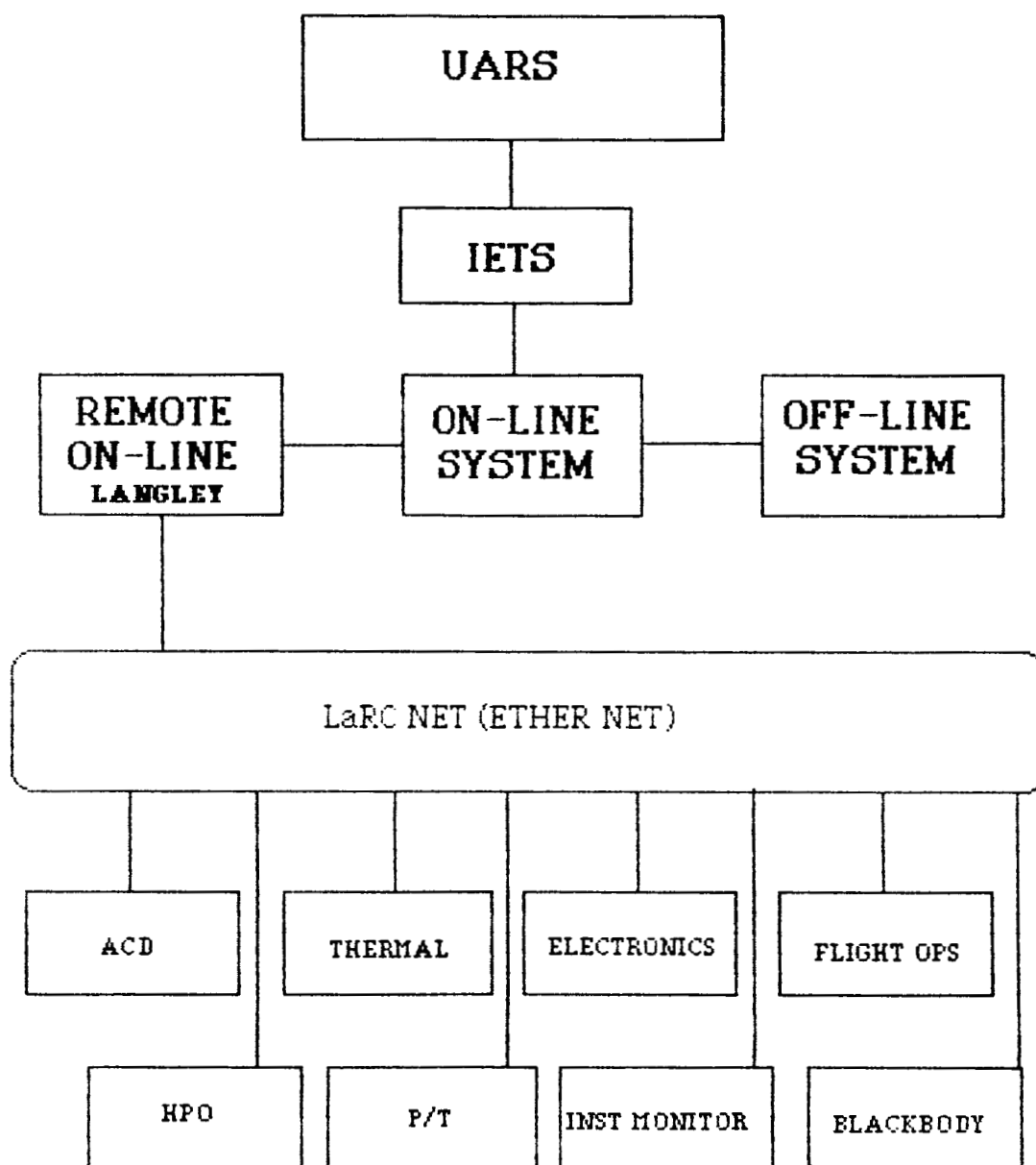
# HALOE QUICK-LOOK DATA SYSTEM LANGLEY TEST SET-UP (PRE I&T)





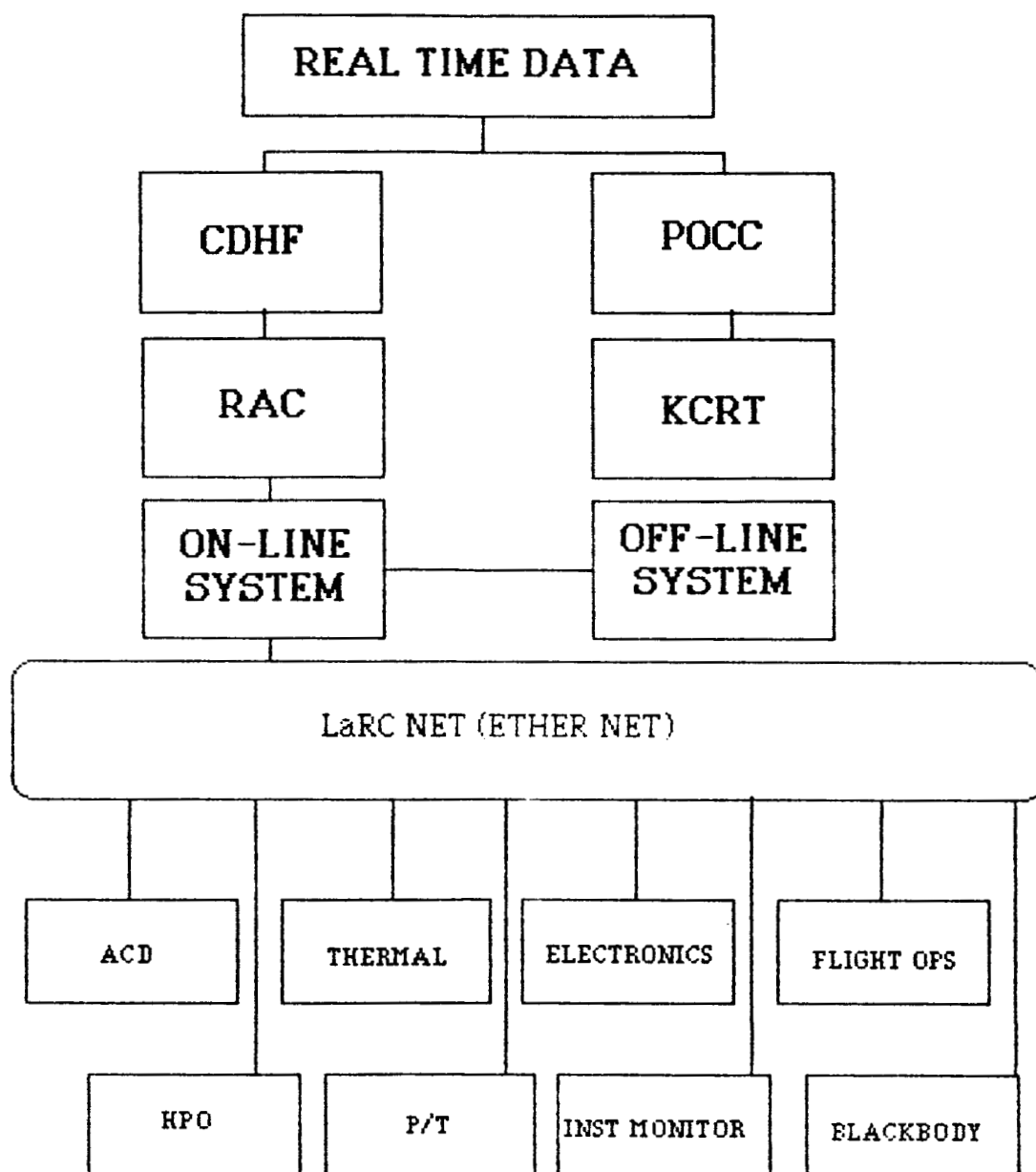
# HALOE QUICK-LOOK DATA SYSTEM

## UARS I & T

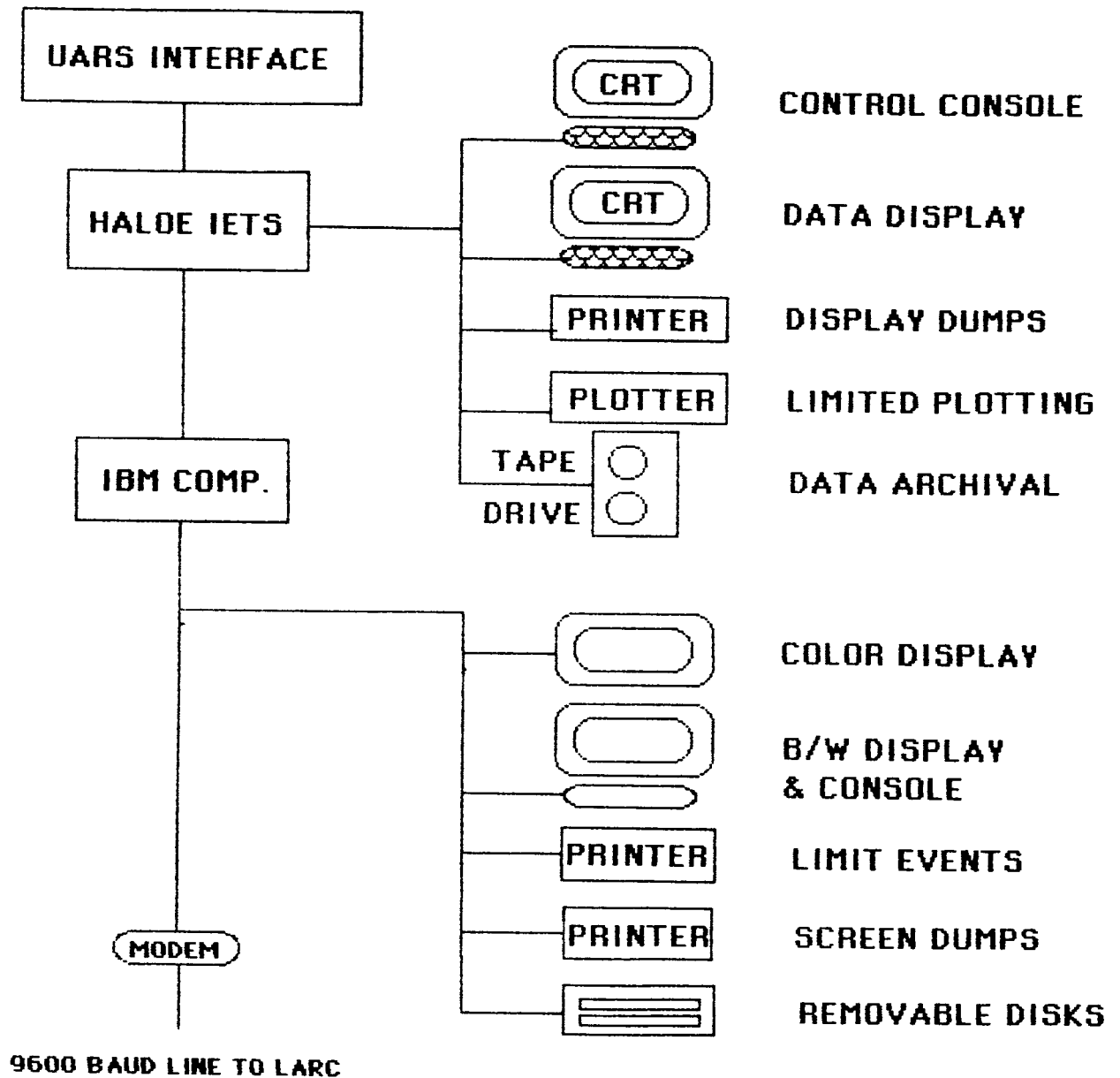


# HALOE QUICK-LOOK DATA SYSTEM

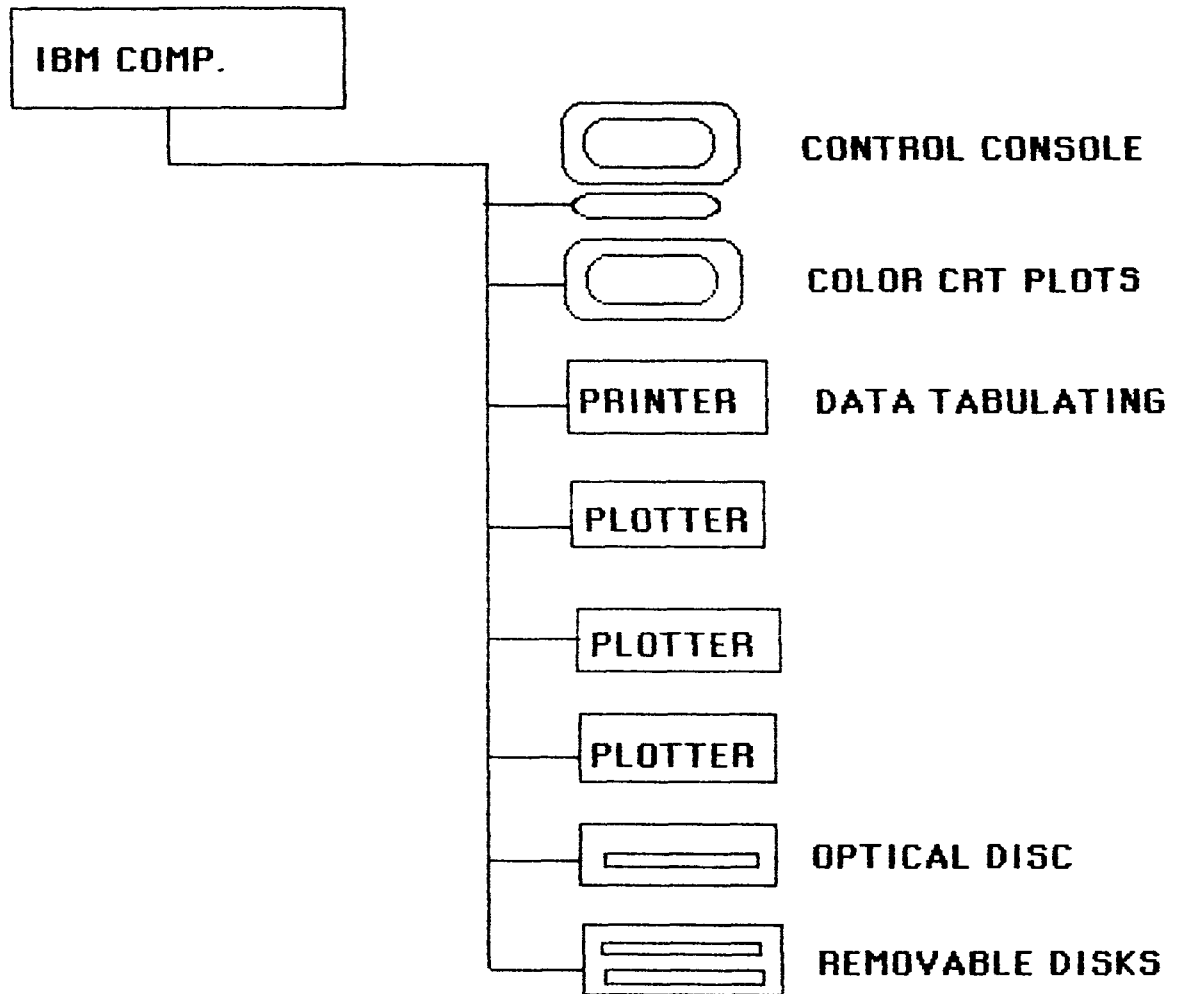
## POST LAUNCH



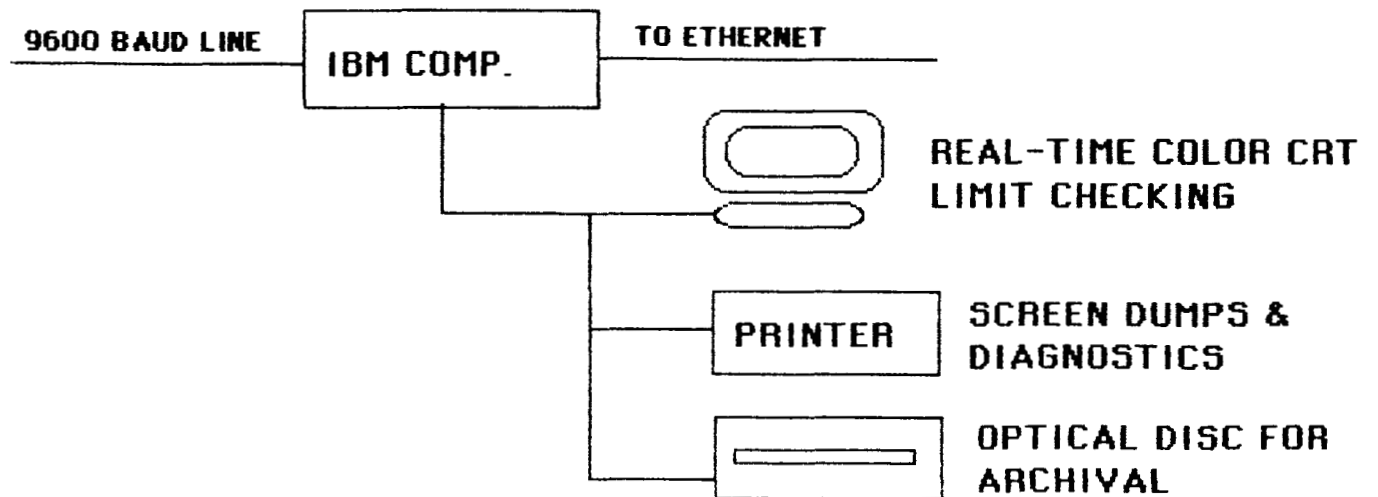
# HALOE/UARS ON-LINE SYSTEM



## HALOE/UARS OFF-LINE SYSTEM



## HALOE - LANGLEY REMOTE ON-LINE DISPLAY



## SECTION 6 - DATA REDUCTION & ANALYSIS

Data reduction and analysis efforts under this contract were largely concerned with the HALOE blackbody life tests. The HPLOT program described elsewhere in this report (and documented in the appendix) was utilized to evaluate, primarily through plot generation, a considerable quantity of HALOE blackbody test data.

HALOE instrument test data tapes were processed using the CDC NOS facility. Utilizing software developed by STX personnel under other contracts, a large number of tapes were converted into data files which were then used to generate a wide variety of plots. These plots were instrumental in the timely evaluation of HALOE EMI and thermal vacuum test data.

## APPENDIX A - HARP

Program Name: HARP (HALOE Analysis and Reduction Program)

Function: HARP is designed to facilitate the processing of HALOE test data tapes for performance verification and characterization of the HALOE instrument.

Description: HARP is a segmented program written in Fortran on an HP-1000 computer. At various stages of development and usage, HARP has had segments which were used to plot parameters on different output devices, to do Fourier analysis and to calculate statistical values such as mean and standard deviation for data taken at different "cal-wheel" positions.

Use: HARP is invoked on an HP-1000 by typing HARP. The program is menu driven and will offer the user flexibility in determining input and output files and plotter devices. The windowing technique offered by HARP greatly facilitates the selection and processing of parameters of interest from the HALOE data stream during times of interest.

```

2 #ENAC(XYZ,0)
3 #FILES(3,3)
4 PROGRAM HARP<>,HALOE ANALYSIS AND REDUCTION PROGRAM <870519.1240>
5 C PROGRAM NAME: HARP
6 C
7 C WRITTEN BY WILLIAM L EDMONDS
8 C STX CORPORATION
9 C NASA EXT 3761
10 C STX 865 0214
11 C
12 C
13 C

```

```

14 C HARP (HALOE ANALYSIS AND REDUCTION PROGRAM) IS THE BASE
15 C SEGMENT OF A SYSTEM OF SOFTWARE DESIGNED TO ANALYZE AND REDUCE
16 C HALOE TEST DATA TAPES. THIS BASE SEGMENT ( REFERRED TO AS HARP)
17 C IS EXECUTED ONLY ONCE. IT CALLS THE MAIN SEGMENT (HARP0) TO
18 C DISPLAY THE OPTION MENU AND PROCESS WHATEVER TASKS THE USER
19 C SELECTS. SEE THE LISTING FOR HARP0 FOR A BRIEF DESCRIPTION
20 C OF ITS FEATURES.
21 C
22 C
23 C
24 C

```

```

25 INTEGER HARP0(3)
26 COMMON/XYZ/ IVDT(7,200), NIBD(500),IVDTN(6),MNE(4,200),X(16384),
27 *NPT(16),IDCNT,IST(6),IET(6),MON(4,16),
28 *IDN(16),ITYP(16),IFREQ(16),XMIN(16),XMAX(16),NPTS
29 *,SUMX(16),SUMX2(16)
30 COMMON LUT,LULOG,LUIN,LUWIN,NTAP,INBUF(10),LBUF(1510),LUPR
31 C
32 C
33 C

```

```

34 CN STRUCTURE OF VARIABLE DEFINITION TABLE (VDT)
35 CN IVDT(I,ID) I=1 TO 7 ID = ID OF ASSOCIATED PARAMETER
36 CN IVDT(1,I) = NIBBLE TABLE POINTER
37 CN IVDT(2,I) = LIMIT TABLE INDEX
38 CN IVDT(3,I) = DESCRIPTION INDEX
39 CN IVDT(4,I) = NUMBER OF OCCURANCES/ MAJOR FRAME
40 CN IVDT(5,I) = START BIT WITHIN NIBBLE
41 CN IVDT(6,I) = LENGTH (BITS)
42 CN IVDT(7,I) = CONVERSION EQUATION #
43 CA
44 CA NIBD(IVDT(1,ID)) - POINTS TO FIRST OCCURANCE OF PARAMETER ID
45 CA NIBD(IVDT(1,ID)+1) TO NIBD(IVDT(1,ID)+IVDT(4,ID)-1) POINT
46 CA TO SUCCESSIVE OCCURANCES OF SAME
47 CA

```

```

48 CA MNE(1,ID) - MNE(4,ID) CONTAINS NAME OF PARAMETER ID
49 CA

```

```

50 C*****
51 COMMON /ENG/ IENG
52 COMMON /IDAT/IBUF(256),IFLAG,IBTIM(6),ISTAT(10),IANHK(24),IPWR(4)
53 *,ITYPE
54 COMMON/LLAGC/LAGC(16)
55 LOGICAL IEQF,LAGC
56 DOUBLE PRECISION*8 XMEAN,VAR,SD,SUMX,SUMX2,DIFF

```

ORIGINAL PAGE IS  
OF POOR QUALITY



```

57 LOGICAL LAGC
58 DATA HARP0/'HARP0 '/'
59 C
60 C
61 C GET INPUT STRING IF ANY
62 C
63 C
64 CALL GETST(INBUF,10,ILOG)
65 IVDTN(1)=2HVA
66 IVDTN(2)=2HRD
67 IVDTN(3)=2HEF
68 IVDTN(4)=2H
69 IVDTN(5)=2H
70 IVDTN(6)=2H
71 CALL LGBUF(LBUF,1510)
72 LUPR=6 ! DEFAULT OUTPUT IS TO PRINTER
73 DO 100 I=1,16
74 100 IFREQ(I)=0
75 CALL SEGLD(HARP0,IRTN)
76 C LOAD MENU SEGMENT HARP0
77 END

```

FTN4X COMPILER: HP92834 REV.2130 (810716)

\*\* NO WARNINGS \*\* NO ERRORS \*\* PROGRAM: 624 COMMON: 1526

ORIGINAL PAGE IS  
OF POOR QUALITY

```
78 BLOCK DATA DLA
79 COMMON LUT,LULOG,LUIN,LUWIN,NTAP,INBUF(10),LBUF(1510),LUPR
80 COMMON/ENG/IENG
81 COMMON /IDAT/IBUF(256),IFLAG,IBTIM(6),ISTAT(10),IANHK(24),IPWR(4)
82 *,ITYPE
83 COMMON/DISP/ IDD(100),IDDS(10),IDDNM(6,6)
84 COMMON/LLAGC/LAGC(16)
85 DATA IENG/2/
86 END
```

FTH4X COMPILER: HP92834 REV.2130 (810716)

\*\* NO WARNINGS \*\* NO ERRORS \*\* PROGRAM: (NONE) COMMON: 1526

```
BLOCK COMMON LLAGC SIZE: 16
BLOCK COMMON DISP SIZE: 146
BLOCK COMMON IDAT SIZE: 302
BLOCK COMMON ENG SIZE: 1
```

ORIGINAL PAGE IS  
OF POOR QUALITY

```

2 $EMA(XYZ,0)
3 $FILES(3,4)
4     PROGRAM HARPO(5,99), MENU SEGMENT FOR HARPO <851010.0905>
5     COMMON /ENG/ IENG
6     COMMON LUT,LULOG,LUIN,LUWIN,NTAP,INBUF(10),LBUF(1510),LUPR
7     COMMON /XYZ/DAT(16384),NPTS(16),IDCNT,IST(6),IET(6),MON(4,16),
8     *IDN(16),ITYP(16),IFREQ(16),XMIN(16),XMAX(16),NPTS
9     *,SUMX(16),SUMX2(16)
10 C
11     COMMON/VDT/IVDT(7,200),NIBD(500),IVDTN(6),MNE(4,200)
12     COMMON/IDAT/IBUF(256),IFLAG,IBTIM(6),ISTAT(10),IANHK(24),IPWR(4)
13     *,ITYPE
14 C
15 C     LU 41 = COMMAND FILE (IF ANY) (LUT)
16 C     LU 8 = TAPE UNIT (LUIN)
17 C     LU 40 = DISK FILE (IF ANY) (LUIN)
18 C     LU 42 = WINDOW FILE (LUWIN)
19 C
20 C
21     LOGICAL IEOF
22     DOUBLE PRECISION*8 XMEAN,VAR,SD,SUMX,SUMX2,DIFF
23 C
24     DIMENSION NAM(6),ISTAR(4),IEND(4),ISTM(6),IETM(6),ITBUF(6),
25     *ITIME(13),ITIMS(13),ITBU2(6),ISCALS(6),IC(16)
26     DIMENSION JTIMS(7),NEMO(4),IDESC(10)
27     INTEGER HARP1(3),HARP2(3)
28     EQUIVALENCE(JTIMS(1),ITIMS(1))
29     INTEGER CKTM
30     DIMENSION MPTS(16),INOTE(38)
31     DATA NAM/'WINDOW':22'/
32     DATA HARP1/'HARP1 '/
33     DATA HARP2/'HARP2 '/
34     DATA ISCALS/'SCALES '/
35
36     DATA ISTAR/'STARTING'/
37     DATA IEND/'ENDING '/
38 C
39 C     HARPO IS THE MENU SEGMENT OF HARPO; GENERAL ANALYSIS PROGRAM
40 C     FOR HALOE. WHEN PROGRAM HARPO IS RUN, THE FIRST SEGMENT LOADED
41 C     WILL BE HARPO. VARIOUS MENU ITEMS CAN THEN BE EXECUTED TO
42 C     SELECT THE INPUT DATA FILE, SELECT A TIME WINDOW, SELECT
43 C     PARAMETERS TO PROCESS AND DETERMINE WHAT CALCULATIONS AND PLOTS
44 C     ARE DESIRED.
45 C
46 C
47     LUT=LOGLU(IDUM) ! GET LU OF TERMINAL
48     LUPR=6
49     NPTS=16384 ! SET DEFAULT NUMBER PTS PER PARAMETER
50     IDCNT=0
51     OPEN(UNIT=20,FILE=IVDTN,Iostat=IOS,ERR=5)
52     CALL RVDT(20)
53     CLOSE(20)
54 C
55 C     READ IN SCALE FACTORS
56 C

```

ORIGINAL PAGE IS  
OF POOR QUALITY

```

57 C
58 OPEN(UNIT=20,FILE=ISCALS,ERR=6,IOSTAT=IOS)
59 CALL RMSCL(20,-1)
60 CLOSE(20)
61 C
62 C
63 C
64 GOTO10
65 5 WRITE(LULOG,2100)IOS,IYDTN
66 STOP
67 6 WRITE(LULOG,2100)IOS,ISCALS
68 STOP
69 10 CONTINUE
70 IF(ILOG.NE.0)THEN ! SEE IF WE'RE USING A DISK COMMAND FILE
71 LUT=41 ! YES...USE LU 41 (ARBITRARY #)
72 OPEN(LUT,IOSTAT=IOS,ERR=1999,FILE=INBUF)
73 ENDIF
74 LULOG=LOGLU(IDUM) ! SET OUTPUT TO TERMINAL
75 WRITE(LULOG,2009) ! DO YOU WANT TO SELECT INPUT FILE?
76 2009 FORMAT(" DO YOU WANT TO SELECT AN INPUT FILE? Y/N")
77 READ(LUT,2001)IANS
78 IF(IANS.EQ.1HY)GOTO100 ! IF YES, GO TO FILE SELECTION ROUTINE
79 C
80 C
81 C DISPLAY MENU AND INPUT SELECTION
82 C
83 C
84 1 WRITE(LULOG,2000)
85 2000 FORMAT("// " 1 = SELECT NEW INPUT FILE NAME OR UNIT "/"
86 * " 2 = SELECT TIME WINDOW "/"
87 * " 3 = SELECT PARAMETERS "/"
88 * " 4 = GENERATE PLOTS "/"
89 * " 5 = TIME SERIES ANALYSIS "/"
90 * " 6 = SEARCH ANNOTATE RECORDS "/"
91 * " 7 = PRINT SELECTED PARAMETERS "/"
92 * " 8 = STATISTICS "/"
93 * " 9 = PROCESS BY PARAMETER VALUE"/
94 * " 10 = EXECUTE A COMMAND FILE",/,
95 * " 11 = SPECIFY OUTPUT LOG DEVICE LU",/,
96 * " 12 = MAKE TREND SNAP-SHOT ",/,
97 * " 13 = QUIT ")
98 14 READ(LUT,*,END=15)IANS
99 GOTO(100,200,300,400,500,600,700,800,900,1000,1100,1200,
100 *1300)IANS
101 15 CLOSE(LUT)
102 LUT=LOGLU(IDUM)
103 GOTO14
104 C
105 C
106 C SELECT INPUT FILE NAME OR UNIT
107 C
108 C
109 100 WRITE(LULOG,2010) ! CHOOSE DISK OR TAPE INPUT
110 2010 FORMAT(" ENTER T FOR TAPE OR D FOR DISK INPUT FILE ")
111 READ(LUT,2001)IANS

```

ORIGINAL PAGE IS  
OF POOR QUALITY

```

112 2001  FORMAT(A1)
113          DO 101 I=1,6
114 101    IBTIM(I)=0
115          IF(IANS.NE.1HT.AND.IANS.NE.1HD)THEN
116          WRITE(LULOG,2002)
117          GOTO1
118          ENDIF
119 2002  FORMAT(" INCORRECT RESPONSE ")
120          CLOSE(LUIN)      ! CLOSE WHATEVER WAS OPEN IF ANYTHING
121          CLOSE(LUWIN)     ! CLOSE WHATEVER WINDOW FILE WAS OPEN
122          IF(IANS.EQ.1HT) THEN
123          LUIN=8           ! INPUT WILL COME FROM TAPE UNIT
124          WRITE(LULOG,2005)
125 2005  FORMAT(" DO YOU WANT TO USE THE ALTERNATE TAPE DRIVE? (Y/N)")
126          READ(LUT,2001)IANS
127          IF(IANS.EQ.1HY)LUIN=9
128          NTAP=5          ! SET FLAG TO FORCE READ BY REDAT ON 1ST CALL
129          OPEN(LUIN,IOSTAT=IOS,ERR=1998)
130          LUWIN = LUIN    ! DEFAULT WINDOW FILE IS THE INPUT FILE
131          ELSE
132  C
133  C      GET NAME OF INPUT DISK FILE
134  C
135          WRITE(LULOG,2003)
136 2003  FORMAT(" ENTER NAME OF INPUT FILE (6A2) ")
137          READ(LUT,2004)NAM
138 2004  FORMAT(6A2)
139          LUIN=40          ! ARBITRARY UNIT NUMBER
140          OPEN(LUIN,IOSTAT=IOS,ERR=1997,FILE=NAME)
141          LUWIN=LUIN      ! DEFAULT WINDOW FILE IS INPUT FILE
142          ENDIF
143          GOTO1            ! END OF OPTION 1
144  C
145  C
146  C
147  C -----
148  C
149  C
150  C      SELECT TIME WINDOW AND CREATE WINDOW FILE
151  C
152  C
153  C
154 200  CONTINUE
155          WRITE(LULOG,2019)
156 2019  FORMAT(" REWIND THE INPUT FILE? Y/N")
157          READ(LUT,2001)IANS
158          IF(IANS.EQ.1HY)REWIND(LUIN)
159 201  WRITE(LULOG,2020)
160 2020  FORMAT(" DO YOU WANT TO SPECIFY START & STOP TIMES (Y/N)")
161          READ(LUT,2001)IANS
162          IF(IANS.EQ.1HN)GOTO250      ! PROCESS FROM CURRENT TIME
163 204  CONTINUE
164          CALL GETIM(LUT,LULOG,ISTAR,ISTM,IER)
165          IF(IER.EQ.0)GOTO205
166 203  WRITE(LULOG,2021)

```

ORIGINAL PAGE IS  
OF POOR QUALITY

```

167 2021 FORMAT(" DO YOU WANT TO RE-ENTER (Y/N)?")
168 READ(LUT,2001)IANS
169 IF(IANS.EQ.1HY)GOTO204
170 GOTO1 ! ABORT THIS OPTION
171 205 CONTINUE
172 CALL GETIM(LUT,LULOG,IEND,IETM,IER)
173 C
174 C NOW PUT START AND STOP TIMES INTO EMA COMMON ARRAYS IST & IET
175 C
176 DO 206 I=1,6
177 IST(I)=ISTM(I)
178 206 IET(I)=IETM(I)
179 CALL REDAT(IEOF,1) ! READ FIRST RECORD
180 IF(IEOF)THEN
181 WRITE(LULOG,2032)
182 GOTO1
183 ENDIF
184 IF(IER.EQ.0)GOTO280
185 WRITE(LULOG,2021)
186 READ(LUT,2001)IANS
187 IF(IANS.EQ.1HY)GOTO205
188 GOTO1 ! ABORT
189 250 WRITE(LULOG,2025)
190 2025 FORMAT(" DO YOU WANT TO EXTRACT DATA STARTING AT ",/,
191 * " CURRENT POSITION OF INPUT FILE? (Y/N)")
192 READ(LUT,2001)IANS
193 IF(IANS.EQ.1HN)GOTO1 ! ABORT
194 DO 252 I=1,6
195 252 ISTM(I)=IBTIM(I)
196 WRITE(LULOG,2026)
197 2026 FORMAT(" ENTER NUMBER OF HOURS,MINUTES & SECS TO PROCESS",/,
198 * " IN THE FORM HH,MM,SS (THREE INTEGERS SEPERATED BY COMMAS)")
199 C
200 READ(LUT,*)IHR,MN,ISEC
201 CALL REDAT(IEOF,1) ! READ FIRST RECORD
202 IF(IEOF)THEN
203 WRITE(LULOG,2032)
204 2032 FORMAT(" INPUT FILE AT EOF, ABORTING ")
205 GOTO1
206 ENDIF
207 DO 260 I=1,6
208 260 ISTM(I)=IBTIM(I)
209 SEC=ISEC
210 CALL ADTIM(ISTM,IHR,MN, SEC,IETM) ! CALCULATE ENDING TIME
211 C
212 WRITE(LULOG,2029)
213 2029 FORMAT(" START, STOP TIMES : ",//)
214 C
215 CALL CNVTM(ISTM,ITIME)
216 WRITE(LULOG,2036)ITIME
217 CALL CNVTM(IETM,ITIME)
218 WRITE(LULOG,2036)ITIME
219 2036 FORMAT(2X,13A2)
220 280 CONTINUE
221 285 WRITE(LULOG,2030)

```

ORIGINAL PAGE IS  
OF POOR QUALITY.

```

222 2030  FORMAT(" DO YOU WANT TO SPECIFY NAME OF WINDOW FILE(Y/N)")
223      READ(LUT,2001)IANS
224      IF(IANS.EQ.1HN)GOTO288
225      WRITE(LULOG,2031)
226 2031  FORMAT(" ENTER WINDOW FILE NAME (6A2)")
227      READ(LUT,2004)NAM
228      LUWIN=0
229 288   IF(LUWIN.EQ.42)THEN
230      WRITE(LULOG,2037)
231 2037  FORMAT(" APPEND TO WINDOW FILE IN USE? Y/N ")
232      READ(LUT,2001)IANS
233      IF(IANS.EQ.1HY)GOTO289
234      CLOSE(LUWIN)
235      ELSE
236      LUWIN=42      ! IN ANY EVENT, A NEW WINDOW FILE IS LU 42
237      OPEN(LUWIN,IOSTAT=IOS,ERR=299,FILE=NAM,STATUS='UNKNOWN')
238      ENDIF
239 C
240 289   CALL SEEK(ISTM,IERR)
241      IF(IERR.GT.0)GOTO299
242 286   CALL REDAT(IEOF,0)      ! ZERO INDICATES ALL RECORD TYPES
243      IF(IEOF)GOTO295
244      IF(CKTM(IBTIM,IETM))287,287,295
245 287   WRITE(LUWIN,ERR=299)ITYPE,IPWR,IBTIM,IBUF,IDUM,IANHK,ISTAT
246      WRITE(LULOG,2049)
247 2049  FORMAT(" STORING DATA IN WINDOW FILE")
248      GOTO286
249 295   REWIND(LUWIN)
250      GOTO1
251 299   WRITE(LULOG,2035)IERR,LUWIN
252 2035  FORMAT(" ERROR# ",I5," ON LU# ",I5)
253      GOTO1
254 C
255 C-----
256 300   CONTINUE
257 C
258 C     SELECT PARAMETERS TO PROCESS
259 C
260      MAXP=16
261      CALL PRAMS(MAXP,IER)
262      CALL XTRAC(8)      ! EXTRACT SELECTED VALUES
263      IF(IER.NE.0)GOTO1
264 C
265 C     INSERT DISPLAY OF PARAMETERS CHOSEN HERE..
266 C
267      GOTO1
268 C
269 C-----
270 C
271 C
272 C
273 400   CONTINUE
274 C
275 C     PLOT SELECTED PARAMETERS
276 C

```

ORIGINAL PAGE IS  
OF POOR QUALITY.

```

277 C
278 CALL SEGLO(HARP1,IERR)
279 GOTO1
280 500 CONTINUE
281 CALL SEGLO(HARP2,IERR)
282 IF(IERR.NE.0)WRITE(LULOG,501)IERR
283 501 FORMAT(" ERROR SCHEDULING HARP2 SEGMENT, ERR#=",I5)
284 GOTO1
285 600 CONTINUE
286 WRITE(LULOG,6001)
287 6001 FORMAT(" FORWARD OR REVERSE SEARCH? (F/R)")
288 READ(LUT,2001)IANS
289 IF(IANS.EQ.1HR)GOTO6500
290 IF(IANS.NE.1HF)THEN
291 WRITE(LULOG,6002)
292 6002 FORMAT(" INVALID RESPONSE!")
293 GOTO1
294 ENDIF
295 601 READ(LUIN,END=6099,ERR=6098)ITYPE,(INBUF(I),I=1,4),IBTIM
296 CALL CNVTM(IBTIM,ITIME)
297 WRITE(LULOG,6003)ITIME
298 IF(IFBRK(KK))1,602,1
299 602 IF(ITYPE.NE.3)GOTO601
300 BACKSPACE(LUIN)
301 READ(LUIN)ITYPE,(INBUF(I),I=1,4),IBTIM,INOTE
302 CALL CNVTM(IBTIM,ITIME)
303 WRITE(LULOG,6003)ITIME,INOTE
304 GOTO601
305 6099 WRITE(LULOG,(' " END OF INPUT FILE"'))
306 GOTO1
307 6098 WRITE(LULOG,(' " ERROR ON INPUT FILE"'))
308 GOTO1
309 6500 BACKSPACE(LUIN)
310 BACKSPACE(LUIN)
311 6501 READ(LUIN,END=6099,ERR=6098)ITYPE,(INBUF(I),I=1,4),IBTIM,INOTE
312 IF(IFBRK(KK))1,6502,1
313 6502 IF(ITYPE.NE.3)GOTO6500
314 CALL CNVTM(IBTIM,ITIME)
315 WRITE(LULOG,6003)ITIME,INOTE
316 GOTO6500
317 6003 FORMAT(1X,13A2,2X,38A2)
318 700 CONTINUE
319 ISEC=0
320 WRITE(LULOG,7010)
321 7010 FORMAT(/," 1 = SELECT PRINT FREQUENCY ",/,
322 *" 2 = PRINT SELECTED PARAMETERS ",/,
323 *" 3 = PRINT IN SELECTED DISPLAY FORMAT",/,
324 *" 4 = RETURN TO MAIN MENU")
325 701 READ(LUT,* )IANS
326 IF(ICHK(IANS,1,4))701,702,701
327 702 GOTO(7100,7200,7300,1)IANS
328 7100 WRITE(LULOG,7011)
329 7011 FORMAT(" ENTER PRINT FREQUENCY ",/,
330 *" 1 = EVERY SECOND",/,
331 *" 2 = EVERY 2 SECONDS...ETC.")

```

ORIGINAL PAGE IS  
OF POOR QUALITY



```

332 READ(LUT,*)ITDEL
333 GOTO700
334 7200 CONTINUE
335 MAXFRQ=1
336 MAXP=16
337 CALL PRAMS(MAXP,IER)
338 CALL XTRAC(MAXFRQ)
339 DO 706 KK=1,IDCNT
340 706 IC(KK)=1
341 DO 703 KK=1,6
342 ITBUF(KK)=IST(KK)
343 703 ITBU2(KK)=IET(KK)
344 CALL CNVTM(ITBUF,ITIMS)
345 CALL CNVTM(ITBU2,ITIME)
346 704 WRITE(LUPR,7000) ITIMS,
347 *((MON(KK,LL),KK=1,4),LL=1,IDCNT)
348 ILINE=0
349 705 ILINE=ILINE+1
350 IF(ILINE.GT.50)GOTO704
351 IHR=0
352 MN=0
353 SEC=FLOAT(ISEC)*1.024
354 CALL ADTIM(ITBUF,IHR,MN, SEC,ITBU2)
355 CALL CNVTM(ITBU2,ITIMS)
356 WRITE(LUPR,7001)JTIMS,(DAT(IND(IC(NP),NP)),NP=1,IDCNT)
357 7000 FORMAT(1H1,/,27X,13A2,/,14X,16(2X,4A2))
358 7001 FORMAT(1X,7A2,1X,16E10.4)
359 DO 710 KK=1,IDCNT
360 IC(KK)=IC(KK)+ITDEL*MAXFRQ
361 IF(IC(KK).GT.NPT(KK))GOTO1
362 710 CONTINUE
363 ISEC=ISEC+ITDEL
364 IF(IFBRK(KL))1,705,1
365 7300 CONTINUE
366 CALL RDISP
367 7301 CONTINUE
368 CALL PRDS(IEOF)
369 IF(IEOF)GOTO1
370 IF(ITDEL.GT.1)CALL SKIPY(LUIN,ITDEL,IEOF,LULOG,NTAP)
371 IF(IEOF)GOTO1
372 IF(IFBRK(KL))1,7301,1
373 C
374 C
375 C
376 800 CONTINUE
377 JFIR=0 ! SET FLAG TO ACQUIRE BEGIN TIME
378 C CALCULATE VARIOUS STATISTICAL VALUES
379 WRITE(LULOG,8000)
380 8000 FORMAT(// " 1 = STATS ON ALL SCIENCE DATA ",/,
381 * " 2 = STATS ON SELECTED PARAMETERS ",/,
382 * " 3 = RETURN TO MAIN MENU")
383 READ(LUT,*)IANS
384 GOTO(8100,8200,1)IANS
385 8100 CONTINUE ! STATS ON ALL SCIENCE DATA
386 IDCNT=12

```

ORIGINAL PAGE IS  
OF POOR QUALITY

```

387 IDN(1)= IDGET(8HNOV )
388 IDN(2)= IDGET(8HNOOV )
389 IDN(3)= IDGET(8HHCLV )
390 IDN(4)= IDGET(8HHCLDV )
391 IDN(5)= IDGET(8HHFV )
392 IDN(6)= IDGET(8HHFDV )
393 IDN(7)= IDGET(8HCH4V )
394 IDN(8)= IDGET(8HCH4DV )
395 IDN(9)= IDGET(8H03V )
396 IDN(10)=IDGET(8HC02V )
397 IDN(11)=IDGET(8HNO2V )
398 IDN(12)=IDGET(8HH20V )
399 DO 807 I=1,12
400 CALL IDMOV(I)
401 IFREQ(I)=8
402 MPTS(I)=0
403 807 CONTINUE
404 GOTO808
405 8200 CALL PRAMS(16,IER)
406 IF(IER.NE.0)GOTO1
407 808 CONTINUE
408 DO 809 I=1,IDCNT
409 SUMX(I)=0.
410 SUMX(I)=0.0
411 XMIN(I)=1.0E20
412 XMAX(I)=-1.E20
413 MPTS(I)=0
414 809 CONTINUE
415 810 CALL REDAT(IEOF,1)
416 IF(IEOF)GOTO820
417 IF(JFIR.EQ.0)THEN
418 JFIR=1
419 DO 817 K=1,6
420 817 ISTM(K)=IBTIM(K)
421 ENDIF ! ACQUIRE BEGINNING TIME
422 DO 815 K=1,IDCNT
423 DO 816 L=1,IFREQ(K)
424 ID=IDN(K)
425 ICNTR=0
426 IDAT=IGET(ID,L,ICNTR,V)
427 SUMX(K)=SUMX(K)+V
428 SUMX2(K)=SUMX2(K)+V*V
429 IF(V.LT.XMIN(K))XMIN(K)=V
430 IF(V.GT.XMAX(K))XMAX(K)=V
431 816 CONTINUE
432 MPTS(K)=MPTS(K)+IFREQ(K)
433 815 CONTINUE
434 C NPTS=NPTS+8 ! NUMBER OF POINTS SUMMED SO FAR
435 IF(IFBRK(KK))820,810,820
436 820 CONTINUE
437 IF(NPTS.EQ.0)THEN
438 WRITE(LULOG,8005)
439 8005 FORMAT(" NO DATA OR EOF ENCOUNTERED IN INPUT FILE")
440 GOTO1
441 ENDIF

```

ORIGINAL PAGE IS  
OF POOR QUALITY

```

442      DO 821 K=1,6
443 821    IETM(K)=IBTIM(K)
444      CALL CNVTM(ISTM,ITIMS)
445      CALL CNVTM(IETM,ITIME)
446      WRITE(LULOG,8001)ITIMS,ITIME
447 8001    FORMAT(// " START : ",13A2,5X," STOP : ",13A2,/)
448      IF(LUPR.NE.0)THEN
449        WRITE(LUPR,'(1H1)')
450        WRITE(LUPR,8001)ITIMS,ITIME
451        WRITE(LUPR,8002)
452      ENDIF
453      WRITE(LULOG,8002)
454 8002    FORMAT( // " NAME          MINIMUM    MAXIMUM    MAX-MIN",
455      * "      MEAN      VARIANCE      STD DEV    #PTS      "// )
456      DO 830 I=1,IDCNT
457        PTS=FLOAT(MPTS(I))
458        XMEAN=SUMX(I)/PTS
459        VAR=(PTS*SUMX2(I)-SUMX(I)*SUMX(I))/((PTS-1.D0)*PTS)
460        IF(VAR.GT.0.0)SD=DSQRT(VAR)
461        DIFF=XMAX(I)-XMIN(I)
462        IF(DIFF.EQ.0.0)THEN
463          VAR=0.0
464          SD=0.0
465        ENDIF
466        WRITE(LULOG,8003)(MON(JJ,I),JJ=1,4),XMIN(I),XMAX(I),DIFF,XMEAN
467      * ,VAR,SD,MPTS(I)
468        IF(LUPR.NE.0)WRITE(LUPR,8003)(MON(JJ,I),JJ=1,4),XMIN(I),XMAX(I),
469      * DIFF,XMEAN,VAR,SD,MPTS(I)
470 8003    FORMAT(1X,4A2,5(E10.6,1X),E10.6,I6)
471 830    CONTINUE
472      GOTO1
473 898    CONTINUE
474      WRITE(LULOG,8004)IOS
475 8004    FORMAT(" ERROR # ",I5," ON WINDOW FILE ")
476      GOTO1
477 900    CONTINUE
478      WRITE(LULOG,9001)
479 9001    FORMAT(" DO YOU WANT TO PROCESS BY PARAMETER VALUE?",
480      * /," (FOR CAL-WHEEL, IFOV, SPECTRAL RESPONSE ETC.)Y/N?")
481      READ(LUT,2001)IANS
482      IF(IANS.EQ.1HN)GOTO1
483 901    CALL GETIM(LUT,LULOG,ISTAR,ISTM,IER)
484      IF(IER.NE.0)THEN
485        WRITE(LULOG,2021)
486        READ(LUT,2001)IANS
487        IF(IANS.EQ.1HN)GOTO1
488        GOTO901
489      ENDIF
490      CALL REDAT(IEOF,1)
491      IF(IEOF)THEN
492        WRITE(LULOG,2032)
493        GOTO1
494      ENDIF
495      CALL SEEK(ISTM,IERR)
496      IF(IERR.NE.0)GOTO9099

```

ORIGINAL PAGE IS  
OF POOR QUALITY

```

497      MAXP=23
498      CALL PRAMS(MAXP,IER)
499      IF( IER.NE.0)GOTO1
500      WRITE(LULOG,9003)
501 9003   FORMAT(" ENTER NAME OF PARAMETER FOR STUDY",/,
502          *"CS3 FOR CAL WHEEL; STATUS2 FOR IFOV,SPECTRAL RESPONSE")
503      READ(LUT,2004)NEMO
504      IDNUM=IDGET(NEMO)
505      IF(IDNUM)910,910,920
506 910    WRITE(LULOG,9004)
507 9004   FORMAT(" NOT WHAT I WAS LOOKING FIR...")
508      GOTO1
509 920    DO 930 I=1,IDCNT
510      NPAR=I
511      IF(IDNUM.EQ.IDN(I))GOTO950
512 930    CONTINUE
513      IDCNT=IDCNT+1
514      IDN(IDCNT)=IDNUM
515      DO 932 I=1,4
516      MON(K,IDCNT)=NEMO(I)
517 932    CONTINUE
518      NPAR=IDCNT
519 950    CONTINUE
520      WRITE(LULOG,9010)
521 9010   FORMAT(" ENTER SHORT DESCRIPTIVE NAME FOR PARAMETER",/,
522          *" SUCH AS: SLIT POSITION OR WAVENUMBER OR CAL POSITION ETC.")
523      READ(LUT,9011>IDESC
524 9011   FORMAT(10A2)
525      WRITE(LULOG,9012)
526 9012   FORMAT(" ENTER # OF SECONDS (MAJOR FRAMES) OF DATA TO ",/,
527          *" PROCESS AT EACH LEVEL OF THE PARAMETER")
528      READ(LUT,*)NFRAM
529      WRITE(LULOG,9013)
530 9013   FORMAT(" ENTER MINIMUM # SECONDS ACCEPTIBLE AT EACH LEVEL")
531      READ(LUT,*)MINF
532      WRITE(LULOG,9014)
533 9014   FORMAT(" ENTER MAXIMUM # LEVELS TO PROCESS")
534      READ(LUT,*)MVAL
535      CALL PMET(NFRAM,MINF,NPAR,MVAL,LUPR,IDESC,ISTN)
536      GOTO1
537 9099   WRITE(LULOG,9002)IERR
538      GOTO1
539 9002   FORMAT(" ERROR #",I5)
540 1000   CONTINUE
541      WRITE(LULOG,1001)
542 1001   FORMAT(" DO YOU WANT TO EXECUTE A COMMAND FILE? (Y/N)")
543      READ(LUT,2001)IANS
544      IF(IANS.EQ.1HN)GOTO1
545      WRITE(LULOG,1002)
546 1002   FORMAT(" ENTER NAME OF COMMAND FILE")
547      READ(LUT,2004)NAM
548      CLOSE(LUT)
549      LUT=41
550      OPEN(LUT,Iostat=IOS,ERR=1999,FILE=NAM)
551      GOTO1

```

ORIGINAL PAGE IS  
OF POOR QUALITY

```

552 1100 CONTINUE
553 C WRITE(LULOG,1101)
554 C1101 FORMAT(" DO YOU WANT TO CHANGE THE LIST LU? (Y/N)")
555 C READ(LUT,2001)IANS
556 C IF(IANS.EQ.1HN)GOTO1
557 C WRITE(LULOG,1102)
558 C1102 FORMAT(" ENTER LU (6=PRINTER, 1 OR 12 = SCREEN, 0 = NONE")
559 C READ(LUT,*)LULOG
560 C GOTO1
561 1200 CONTINUE ! TREND SNAP-SHOT
562 WRITE(LULOG,1201)
563 1201 FORMAT(" DO YOU WANT TO SAVE A SNAP-SHOT? Y/N")
564 READ(LUT,2001)IANS
565 IF(IANS.NE.1HY)GOTO1
566 WRITE(LULOG,1202)
567 1202 FORMAT("ENTER TREND FILE NAME")
568 READ(LUT,2004)NAM
569 OPEN(UNIT=20, IOSTAT=IOS,ERR=1299,FILE=NAM)
570 1203 READ(20,ERR=1299,END=1204)
571 GOTO1203
572 1204 WRITE(20,ERR=1299)ITYPE,IPWR,IBTIM,IBUF,IDUM,IANHK,ISTAT
573 CLOSE(20)
574 GOTO1
575 1299 WRITE(LULOG,1298)IOS,NAM
576 1298 FORMAT(" ERROR # ",I5," ON FILE ",6A2)
577 CLOSE(20)
578 GOTO1
579 1300 STOP
580 1997 LUT=LOGLU(IDUM) ! RESET LUT TO TERMINAL
581 WRITE(LULOG,2100)IOS,NAM
582 GOTO1
583 1998 LUT=LOGLU(IDUM)
584 WRITE(LULOG,2101)IOS
585 GOTO1
586 1999 LUT=LOGLU(IDUM)
587 WRITE(LULOG,2102)IOS
588 GOTO1
589 2100 FORMAT(" ERROR # ",I5,2X," FILE NAME :",6A2)
590 2101 FORMAT(" ERROR # ",I5,2X," WITH MAG TAPE ")
591 2102 FORMAT(" ERROR # ",I5," WITH COMMAND FILE ")
592 END

```

FTN4X COMPILER: HP92834 REV.2130 (810716)

\*\* NO WARNINGS \*\* NO ERRORS \*\* PROGRAM: 4881 COMMON: 1526

ORIGINAL PAGE IS  
OF POOR QUALITY

```

593 #EMAC(XYZ,0)
594 SUBROUTINE PMET(NFRAM,MINF,NPAR,MVAL,IPRT,IDESC,ISTM)
595 COMMON/IDAT/IBUF(256),IFLAG,IBTIM(6),ISTAT(10),IANHK(24),IPWR(4)
596 COMMON LUT,LULOG,LUIN,LUWIN,NTAP,INBUF(10),LBUF(1510),LUPR
597 COMMON /XYZ/DAT(16384 ),NPT(16),IDCNT,IST(6),IET(6),MON( 4,16),
598 *IDN(16),ITYP(16),IFREQ(16),SUMX2(16),NPTS
599 *,SUMX(16),XMEAN(16)
600 DIMENSION IDESC(10),ISTM(6),ITBUF(13),PMEAN(16)
601 LOGICAL IEOF
602 DOUBLE PRECISION*8 XMEAN,VAR,SD,SUMX,SUMX2,DIFF
603 C
604 DIMENSION MPTS(24),NAMP(6)
605 C
606 C
607 C THIS ROUTINE PROCESSES DATA AT TIMES WHEN SOME VALUE SUCH
608 C AS CAL WHEEL POSITION IS CONSTANT. SLIT POSITION OR SPECTRAL
609 C WAVELENGTH ARE TWO OTHER TYPES OF PARAMETERS WHICH CAN BE
610 C PROCESSED WITH THIS ROUTINE.
611 C
612 C NFRAM = DESIRED NUMBER OF FRAMES OF DATA AT EACH LEVEL OR
613 C VALUE OF PARAMETER (CAL WHEEL POSITION ETC.)
614 C MINF = MINIMUM NUMBER OF FRAMES ACCEPTIBLE AT EACH LEVEL
615 C NPAR = ID NUMBER OF PARAMETER BEING STUDIED
616 C MVAL = MAXIMUM NUMBER OF LEVELS TO STUDY
617 C IPRT = PRINT FLAG (0= NO PRINTOUT, OTHERWISE PRINT)
618 C
619 C
620
621 CALL CNVTM(ISTM,ITBUF)
622 WRITE(IPRT,1102)ITBUF,IDESC
623 1102 FORMAT(1H1,15X,13A2,10X,10A2)
624 WRITE(LULOG,1103)
625 1103 FORMAT(" DO YOU WANT TO CREATE A PLOT FILE? Y/N")
626 READ(LUT,1104)IANS
627 1104 FORMAT(A1)
628 IF(IANS.EQ.1HY)THEN
629 WRITE(LULOG,1105)
630 1105 FORMAT(" ENTER NAME OF PLOT FILE")
631 READ(LUT,1106)NAMP
632 1106 FORMAT(6A2)
633 IPFLAG=1
634 OPEN(20,FILE=NAMP,ERR=1120)
635 BTIM= ISTM(2)+ISTM(3)*60.+ISTM(4)*3600.
636 WRITE(20,1121)IDCNT,ISTM(6),ISTM(5),BTIM,((MON(I,J),I=1,4),J=1
637 *,IDCNT)
638 1121 FORMAT(13,2I5,F10.3,6A2)
639 ELSE
640 IPFLAG=0
641 ENDIF
642 WRITE(IPRT,1100)((MON(KK,I),KK=1,4),I=1,IDCNT)
643 1100 FORMAT( //,4X,7(4A2,10X))
644 NVAL=0
645 DO 5 KL=1,6
646 IST(KL)=IBTIM(KL)
647 5 CONTINUE

```

```

648 1      IF(NVAL.EQ.MVAL) GOTO230
649      NVAL = NVAL +1
650      IFRAM=0      ! LOCAL COUNTER FOR # FRAMES AT CURRENT LEVEL
651      DO 10 I=1,IDCNT
652      SUMX(I)=0.      ! INITIALIZE SUM TO ZERO
653      SUMX2(I)=0.      ! INITIALIZE SUM X SQUARED TO 0.
654      XMEAN(I) =0.      ! INIT SUM OF SQUARES
655 C      XMIN(I)=1.E20      ! INIT MIN VALUES
656 C      XMAX(I) =-1.E20      ! INIT MAX VALUES
657      MPTS(I)=0      ! INIT NUMBER OF PTS FOR EACH ID
658 10      CONTINUE
659      ICNTR=0
660      IPAR=IDN(NPAR)
661      IDAT=IGET(IPAR,1,ICNTR,V)
662      VAL=V
663      DAT(IND(NVAL,NPAR))=V      ! GET NPAR PARAMETER
664 20      DO 100 K=1,IDCNT
665 C      IF(IDN(K).EQ.IPAR)GOTO100
666      DO 90 L=1,IFREQ(K)
667      ID=IDN(K)
668      ICNTR=0
669      IDAT=IGET(ID,L,ICNTR,V)
670      SUMX(K)=SUMX(K)+V
671      SUMX2(K)=SUMX2(K)+V*V
672 C      IF(V.LT.XMIN(K))XMIN(K)=V
673 C      IF(V.GT.XMAX(K))XMAX(K)=V
674 90      CONTINUE
675      MPTS(K)=MPTS(K)+IFREQ(K)
676 100      CONTINUE
677      IFRAM=IFRAM+1
678      IF(IFRAM.EQ.NFRAM)GOTO200
679 30      CALL REDAT(IEOF,1)
680      IF(IEOF)THEN
681      WRITE(LULOG,1000)
682 1000      FORMAT(" EOF ENCOUNTERED IN INPUT FILE")
683      GOTO230
684      ENDIF
685      ICNTR=0
686      IDAT=IGET(IPAR,1,ICNTR,V)
687      IF(V.EQ.VAL)GOTO20
688      IF(IFRAM.GT.MINF)GOTO200      ! FINISHED THIS LEVEL
689      WRITE(LULOG,1001)IFRAM,VAL      ! NOT ENOUGH POINTS
690 1001      FORMAT(" FOUND ONLY ",I5," FRAMES AT LEVEL =",E12.4)
691      NVAL =NVAL-1
692      GOTO1
693 200      CONTINUE
694      DO 210 KL=1,6
695      IET(KL)=IBTIM(KL)
696 210      CONTINUE      ! GET ENDING TIME
697      DO 220 I= 1,IDCNT
698 C      IF(IDN(I).EQ.IPAR)GOTO220
699      PTS=FLOAT(MPTS(I))
700      XMEAN(I)=SUMX(I)/PTS
701      PMEAN(I)=XMEAN(I)
702      VAR=(PTS*SUMX2(I)-SUMX(I)*SUMX(I))/((PTS-1.D0)*PTS)

```

```
703      IF(VAR.GT.0.D0)THEN
704      SD=DSQRT(VAR)
705      ELSE
706      SD=0.0
707 C      DIFF=XMAX(I)-XMIN(I)
708 C      IF(DIFF.EQ.0.)THEN
709 C      VAR=0.0
710 C      SD=0.0
711      ENDIF
712      SUMX(I)=SD
713 220      CONTINUE
714      IF(IPFLAG.NE.0)
715      *WRITE(20)BTIM,<PMEAN(K),K=1,IDCNT>
716      IF(LUPR.NE.0)
717      *WRITE(LUPR,1010)<<XMEAN(K),SUMX(K)>>,K=1,IDCNT>
718 1010  FORMAT(1X,14(F9.4))
719 225      CONTINUE
720      CALL REDAT(IEOF,1)
721      IF(IEOF)GOTO230
722      ICNTR=0
723      IDAT=IGET(IPAR,1,ICNTR,V)
724      IF(V.EQ.VAL)GOTO225
725      VAL=V
726      GOTO1
727 230      CLOSE(20)
728      RETURN
729 1120      WRITE(LULOG,1119)
730 1119      FORMAT("ERROR OPENING PLOT FILE")
731      RETURN
732      END
```

FTN4X COMPILER: HP92834 REV.2130 (810716)

\*\* NO WARNINGS \*\* NO ERRORS \*\* PROGRAM: 1220 COMMON: 1526



```

733 SUBROUTINE ADTIM(ISTM,IHR,MN, SEC,IETM)
734 DIMENSION ISTM(6),IETM(6)
735 ISEC=SEC ! TRUNCATE VALUE OF SECONDS
736 RSEC=SEC-FLOAT(ISEC)
737 JSEC=RSEC*100
738 IETM(1) = ISTM(1)+JSEC ! SET ENDING .01 SECS TO STARTING VAL
739 ICARY=IETM(1)/100
740 IETM(1)=MOD(IETM(1),100)
741 IETM(2) = ISTM(2)+ISEC+ICARY !ADD SECONDS TO STARTING SECS
742 ICARY = IETM(2)/60 ! CALCULATE CARRY FOR MINUTES
743 IETM(2) = MOD(IETM(2),60) ! MOD MINUTES TO INSURE < 60
744 IETM(3) = ISTM(3) + MN + ICARY ! CALCULATE MINUTES
745 ICARY = IETM(3)/60 ! CALCULATE CARRY FOR HOURS
746 IETM(3) = MOD(IETM(3),60) ! ADJUST MINUTES < 60
747 IETM(4) = ISTM(4) + IHR + ICARY ! CALCULATE HOURS
748 ICARY = IETM(4)/24 ! CALCULATE CARRY FOR DAYS
749 IETM(4) = MOD(IETM(4),24) ! INSURE THAT HOURS<24
750 IETM(5) = ISTM(5) + ICARY ! CALCULATE ENDING DAY
751 IYMOD = 365 ! SET # DAYS IN YEAR
752 IF(MOD(ISTM(6),4).EQ.0) IYMOD= 366 ! CHECK FOR LEAP YEAR
753 ICARY = IETM(5)/IYMOD ! CALCULATE YEAR CARRY
754 IETM(6) = ISTM(6) + ICARY ! ENDING YEAR
755 RETURN
756 END

```

FTN4X COMPILER: HP92834 REV.2130 (810716)

\*\* NO WARNINGS \*\* NO ERRORS \*\* PROGRAM: 178 COMMON: (NONE)

ORIGINAL PAGE IS  
OF POOR QUALITY

```

757 $EMAC(XYZ,0)
758 SUBROUTINE IDMOV(ID)
759 COMMON /XYZ/DAT(16384 ),NPT(16),IDCNT,IST (6),IET (6),MONK 4,16),
760 *IDN(16),ITYP(16),IFREQ(16),XMIN(16),XMAX(16),NPTS
761 *,SUMX(16),SUMX2(16)
762 COMMON/VDT/IVDT(7,200),NIBD(500),IVDTN(6),MNE(4,200)
763 DO 10 I=1,4
764 10 MON(I,ID)=MNE(I,IDN(ID))
765 RETURN
766 END

```

FTN4X COMPILER: HP92834 REV.2130 (810716)

\*\* NO WARNINGS \*\* NO ERRORS \*\* PROGRAM: 50 COMMON: (NONE)

```

767 SUBROUTINE JULIN(IDAY,IYR,IM,IDA)
768 DIMENSION IMS(12),IDY(13)
769 INTEGER*4 IM,IMS
770 DATA IMS /'JAN FEB MAR APR MAY JUNEJULYAUG SEPTOCT NOV DEC '/
771 DATA IDY /0,31,59,90,120,151,181,212,243,273,304,334,365/
772 IAD = 0
773 IF(IDAY.LT.60)GO TO 5
774 IADD = MOD(IYR,4)
775 IF(IADD.EQ.0)IAD = 1
776 5 DO 10 I=2,13
777 IDC = IDY(I) + IAD
778 IF(IDAY.LE.IDC)GO TO 20
779 10 CONTINUE
780 20 IMN=I-1
781 IDA = IDAY - IDY(IMN)
782 IF(IDY(IMN).GT.31)IDA = IDA - IAD
783 IM = IMS(IMN)
784 RETURN
785 END

```

FTN4X COMPILER: HP92834 REV.2130 (810716)

\*\* NO WARNINGS \*\* NO ERRORS \*\* PROGRAM: 127 COMMON: (NONE)

```

786      SUBROUTINE SKIPY(LUIN,ITDEL,IEOF,LULOG,NTAP)
787 C
788 C      SKIP RECORDS IN THE INPUT FILE
789      LOGICAL IEOF
790      ITER=ITDEL-1
791      DO 10 I=1,ITER
792      READ(LUIN,END=20,ERR=30,IOSTAT=IERR)
793 10      CONTINUE
794      NTAP=5
795      CALL REDAT(IEOF,1)
796      IEOF=.FALSE.
797      RETURN
798 20      IEOF=.TRUE.
799      RETURN
800 30      WRITE(LULOG,1000)IERR
801 1000    FORMAT(" ERROR # ",I5," ON INPUT FILE ")
802      RETURN
803      END

```

FTN4X COMPILER: HP92834 REV.2130 (810716)

\*\* NO WARNINGS \*\* NO ERRORS \*\* PROGRAM: 84 COMMON: (NONE)

ORIGINAL PAGE IS  
OF POOR QUALITY

804 SUBROUTINE PRDS  
805 C DUMMY SUBROUTINE  
806 RETURN  
807 END

FTN4X COMPILER: HP92834 REV.2130 (810716)

\*\* NO WARNINGS \*\* NO ERRORS \*\* PROGRAM: 5 COMMON: (NONE)

```
808      BLOCK DATA HADAT
809      COMMON/IDAT/IBUF(256),IFLAG,IBTIM(6),ISTAT(10),IANHK(24),IPWR(4)
810      *,ITYPE
811      COMMON/ENG/IENG
812      COMMON/DISP/ IDDC(100),IDDS(10),IDDNM(6,6)
813      DATA IENG/2/
814      END
```

FTH4X COMPILER: HP92834 REV.2130 (810716)

\*\* NO WARNINGS \*\* NO ERRORS \*\* PROGRAM: (NONE) COMMON: (NONE)

BLOCK COMMON DISP SIZE: 146

BLOCK COMMON ENG SIZE: 1

BLOCK COMMON IDAT SIZE: 302

ORIGINAL PAGE IS  
OF POOR QUALITY

```

815 SUBROUTINE RDISP, READ DISPLAY FORMAT FILE (WLE)
816 DIMENSION IFILE(6)
817 C COMMON/MONTR/ ITCLS,ITLEN,ISBUF(920)
818 COMMON/VDT/ IVDT(7,200),NIBD(500),IVDTN(6),MNE(4,200)
819 COMMON/DISP/ IDD(100),IDDS(10),IDDNM(6,6)
820 COMMON LUT,LULOG,LUIN,LUWIN,NTAP,INBUF(10),LBUF(1510),LUPR
821 C COMMON/MSK/ MASK(16)
822 C DATA IFILE/' ' ;DS:22'/
823 MXIDD = 100
824 LUBD = 20
825 LUDIR = 21
826 11 CALL FMTDR(LULOG,LUDIR,IDDNM)
827 1 WRITE(LULOG,('( " ENTER DISPLAY FORMAT #: _")')
828 READ(LUT,*,ERR=1)IDN
829 IF(ICHK(IDN,1,7))1,2,1
830 2 CONTINUE
831 IF(IDN.NE.7)GO TO 22
832 CLOSE(LUDIR)
833 WRITE(LULOG,('( " ENTER NAME OF FILE: _")')
834 READ(LUT, '(3A2)')(IFILE(I),I=1,3)
835 GO TO 33
836 22 DO 3 I=1,6
837 3 IFILE(I)=IDDNM(I,IDN)
838 33 CONTINUE
839 OPEN(UNIT=LUBD,FILE=IFILE,IOSTAT=ISTAT,ERR=990)
840 REWIND LUBD
841 4 DO 5 I=1,MXIDD
842 5 READ(LUBD,*,END=6)IDD(I)
843 CLOSE(LUBD)
844 CLOSE(LUDIR)
845 6 RETURN
846 990 WRITE(LULOG,('( " ERROR OPENING ",6A2)')(IDDNM(I,IDN),I=1,6)
847 END

```

FTH4X COMPILER: HP92834 REV.2130 (810716)

\*\* NO WARNINGS \*\* NO ERRORS \*\* PROGRAM: 298 COMMON: 1526

```

848      SUBROUTINE FMTDR(LULOG,LUDIR,IDIR),DISPLAY TLM FORMAT DIRECTORY
849 C      DIMENSION IDIR(6,6),INAM(3)
850 C      DATA INAM/6HFMDIR /
851 C      OPEN(UNIT=LUDIR,FILE=INAM,IOSTAT=ISTAT,ERR=990)
852 C      REWIND LUDIR
853 C      READ(LUDIR)((IDIR(J,K),J=1,6),K=1,6)
854 C      DO 10 K=1,6
855 C        WRITE(LULOG, '(I4,1X,6A2)' )K,(IDIR(J,K),J=1,6)
856 C10    CONTINUE
857 C      WRITE(LULOG, '( " 7 ENTER DISPLAY FILE NAME" )' )
858 C      RETURN
859 990    CONTINUE
860      WRITE(LULOG, '( " ERROR OPENING FMT DIR" )' )
861      RETURN
862      END

```

FTN4X COMPILER: HP92834 REV.2130 (810716)

\*\* NO WARNINGS \*\* NO ERRORS \*\* PROGRAM: 35 COMMON: (NONE)



```

863 #EMA(XYZ,0)
864     SUBROUTINE PRAMS(MAXP,IER)
865     DIMENSION NEMO(4),IHELP(2)
866     COMMON /XYZ/DAT(16384 ),NPT(16),IDCNT,IST(6),IET(6),MON(4,16),
867     *IDN(16),ITYP(16),IFREQ(16),XMIN(16),XMAX(16),NPTS
868     *,SUMX(16),SUMX2(16)
869     COMMON LUT,LULOG,LUIN,LUWIN,NTAP,INBUF(10),LBUF(1510),LUPR
870     COMMON/IDAT/IBUF(256),IFLAG,IBTIM(6),ISTAT(10),IANHK(24),IPWR(4)
871     *,ITYPE
872     COMMON/VDT/IVDT(7,200),NIBD(500),IVDTN(6),MNE(4,200)
873     DATA IHELP/'HELP'/
874     IER=0
875     IF(IDCNT.NE.0)THEN                ! DISPLAY PARAMS ALREADY CHOSEN
876     DO 100 KL=1,IDCNT
877     WRITE(LULOG,1001)(MON(K,KL),K=1,4)
878 100    CONTINUE
879 1001   FORMAT(1X,4A2)
880     NPTS=16384/IDCNT
881     WRITE(LULOG,1002)
882 1002   FORMAT(" THESE ARE THE CURRENT PARAMETERS, DO YOU ",/,
883     *" WISH TO ENTER A NEW SET? (Y/N)")
884     READ(LUT,'(A1)')IANS
885     IF(IANS.EQ.1HN)RETURN
886     ENDIF
887 300    IDCNT=0
888 301    IDCNT=IDCNT+1
889     IF(IDCNT.GT.MAXP)GOTO350
890 305    WRITE(LULOG,3000)
891 3000   FORMAT(" ENTER PARAMETER NAME,HELP OR STOP")
892     READ(LUT,2004)NEMO
893 2004   FORMAT(6A2)
894     IF(NEMO(1).EQ.2HST.AND.NEMO(2).EQ.2HOP)GOTO350
895     IF(NEMO(1).NE.2HHE.OR.NEMO(2).NE.2HLP)GOTO302
896 C
897 C     DISPLAY MNEMONICS HERE....
898 C
899     WRITE(LULOG,3005)MNE
900 3005   FORMAT(/,(9(4A2) ))
901     GOTO305
902 302    IDN(IDCNT)=IDGET(NEMO)
903     IF(IDN(IDCNT))303,350,310
904 303    WRITE(LULOG,3002)
905     GOTO305
906 310    DO 312 K=1,4
907 312    MON(K,IDCNT)=NEMO(K)
908 313    WRITE(LULOG,3001)
909 3001   FORMAT(" ENTER TYPE (1=HEX,2=ENG,3=TEMP)_")
910     READ(LUT,*,ERR=320)ITY
911     IF(ICHK(ITY,1,3))320,325,320
912 320    WRITE(LULOG,3002)
913 3002   FORMAT(" INVALID ")
914     GOTO313
915 325    ITYP(IDCNT)=ITY
916     IFREQ(IDCNT)=IVDT(4,IDN(IDCNT))    ! GET THE FREQ
917     GOTO301

```

```

918 350   IDCNT=IDCNT-1
919       DO 355 KL=1,IDCNT
920       WRITE(LULOG,3030)(MON(K,KL),K=1,4),
921       *IDN(KL),ITYP(KL),NPT(KL)
922 355   CONTINUE
923 3030   FORMAT(1X,4A2,3I5)
924       WRITE(LULOG,3031)
925 3031   FORMAT(" ARE THESE PARAMETERS CORRECT? Y/N ")
926       READ(LUT,'(A1)')IANS
927       IF(IANS.EQ.1HN)GOTO300
928 360   NPTS=16384/IDCNT
929 C     CALL XTRAC             ! EXTRACT THE DESIRED VARIABLES
930 C     CALL TO XTRACT WAS PLACED IN MAIN PROGRAM.
931       RETURN
932       END

```

FTN4X COMPILER: HP92834 REV.2130 (810716)

\*\* NO WARNINGS \*\* NO ERRORS \*\* PROGRAM: 683 COMMON: 1526

933 SUBROUTINE XTRAC(MAXFRQ)  
934 RETURN  
935 END

FTN4X COMPILER: HP92834 REV.2130 (810716)

\*\* NO WARNINGS \*\* NO ERRORS \*\* PROGRAM: 6 COMMON: (NONE)

```

936      SUBROUTINE GETIM(LUT,LULOG,ISTRG,ITIM,IER),PROMPT USER FOR TIME
937 C
938 C
939 C      GETIM PROMPTS THE USERS FOR TIME INPUT.
940 C      FIRST IT ASKS FOR MONTH/DAY/YEAR AND THEN
941 C      IT ASKS FOR HOURS/MIN/SEC. IF NO ERRORS ARE DETECTED
942 C      IT WILL RETURN A VALUE OF ZERO FOR IER. LUT IS THE
943 C      INPUT LOGICAL UNIT, LULOG IS THE LOGICAL UNIT FOR
944 C      DIAGNOSTIC OUTPUT. ISTRG IS A STRING (EITHER "BEGINNING"
945 C      OR "ENDING" USED IN PROMPTING INPUT. ON OUTPUT, ITIM WILL
946 C      CONTAIN:
947 C      ITIM(6) = YEAR (TWO DIGITS E.G. 85)
948 C      ITIM(5) = DAY NUMBER (DAY OF YEAR)
949 C      ITIM(4) = MILITARY HOUR NUMBER (0 TO 23)
950 C      ITIM(3) = MINUTES (0 TO 59)
951 C      ITIM(2) = SECONDS (0 TO 59)
952 C      ITIM(1) = .01 SECONDS (SET TO ZERO IN THIS ROUTINE)
953 C
954      DIMENSION ITIM(6)
955      DIMENSION IDAY(12),ISTRG(4),IMO(12)
956      DATA IDAY/31,28,31,30,31,30,31,31,30,31,30,31/
957      DATA IMO/0,31,59,90,120,151,181,212,243,273,304,334/
958 C
959 C
960 C
961      IER = 1      ! SET ERROR FLAG TO INDICATE ERROR
962      WRITE(LULOG,2200)ISTRG      ! PROMPT USER FOR MN/DA/YR
963 2200  FORMAT(" ENTER ",4A2," TIME: MN/DA/YR ")
964 1      READ(LUT,*,ERR=1)MN,IDA,IYR
965 2201  FORMAT(I2,1X,I2,1X,I2)
966      IF(MN.GT.0.AND.MN.LT.13)GOTO205
967      WRITE(LULOG,2202)
968 2202  FORMAT(" WRONG!")
969      RETURN
970 205    IF(IDA.GT.0.AND.IDA.LE.IDAY(MN))GOTO210
971      IF(MN.EQ.2.AND.AMOD(FLOAT(IYR),4.).EQ.0..AND.IDAY.EQ.29)GOTO210
972      WRITE(LULOG,2203)
973 2203  FORMAT(" INCORRECT DAY # ")
974      RETURN
975 210    IF(IYR.GT.83.AND.IYR.LT.99)GOTO215
976      WRITE(LULOG,2204)
977 2204  FORMAT(" I DON'T THINK THE YEAR IS CORRECT!")
978      RETURN
979 215    WRITE(LULOG,2205)
980 2205  FORMAT(" ENTER HRS:MIN:SECS   E.G. 14:15:00 ( = 2:15 PM)")
981 2      READ(LUT,*,ERR=215)IHR,MIN,ISEC
982      IF(IHR.GE.0.AND.IHR.LT.24)GOTO220
983      WRITE(LULOG,2206)
984 2206  FORMAT(" INVALID ENTRY")
985      RETURN
986 220    IF(MIN.GE.0.AND.MIN.LT.60)GOTO225
987      WRITE(LULOG,2206)
988      RETURN
989 225    IF(ISEC.GE.0.AND.ISEC.LT.60)GOTO230
990      WRITE(LULOG,2206)

```

ORIGINAL PAGE IS  
OF POOR QUALITY

```
991      RETURN
992 230    IER=0      ! SET ERROR FLAG TO NO ERROR STATUS
993      ITIM(1)=0
994      ITIM(2)=ISEC
995      ITIM(3) = MIN
996      ITIM(4) = IHR
997      ITIM(5) = IDA+IMO(MN)
998      IF(MN.GT.2.AND.AMOD(FLOAT(IYR),4.).EQ.0)ITIM(5)=ITIM(5)+1
999      ITIM(6) = IYR+1900
1000     RETURN
1001     END
```

FTN4X COMPILER: HP92834 REV.2130 (810716)

\*\* NO WARNINGS \*\* NO ERRORS \*\* PROGRAM: 443 COMMON: (NONE)

```

1002      SUBROUTINE SEEK(ISTM,IERR),SEEK TIME ON INPUT FILE
1003 C
1004 C
1005 C
1006 C      SUBROUTINE SEEK LOOKS FOR A REQUESTED TIME IN THE INPUT FILE
1007 C
1008 C
1009      DIMENSION ISTM(6),ITIME(13)
1010      COMMON/IDAT/IBUF(256),IFLAG,IBTIM(6),ISTAT(10),IANHK(24),IPWR(4)
1011      *,ITYPE
1012      COMMON LUT,LULOG,LUIN,LUWIN,NTAP,INBUF(10),LBUF(1510),LUPR
1013      LOGICAL IEOF
1014      INTEGER CKTM
1015      IERR=1      ! INITIALIZE FLAG TO ERROR
1016      CALL CNVTM(IBTIM,ITIME)
1017      WRITE(LULOG,1000)ITIME
1018      CALL CNVTM(ISTM,ITIME)
1019      WRITE(LULOG,1001)ITIME
1020 1001  FORMAT(" SEEKING :",13A2)
1021      IF(CKTM(IBTIM,ISTM))100,300,300      ! SEE IF WE'RE ALREADY THERE
1022 100   READ(LUIN,END=103,ERR=900)ITYPE,IPWR,IBTIM
1023      IF(ITYPE.EQ.1)GOTO104
1024      GOTO100
1025 103   WRITE(LULOG,1003)
1026 1003  FORMAT(" EOF ON INPUT FILE, CONTINUE? Y/N ")
1027      READ(LUT,1004)IANS
1028 1004  FORMAT(A1)
1029      IF(IANS.NE.1HY)RETURN
1030      GOTO100
1031 104   CALL CNVTM(IBTIM,ITIME)
1032      WRITE(LULOG,1000)ITIME
1033 1000  FORMAT(" TIME = ",13A2)
1034      IF(IFBRK(KK))900,101,900
1035 101   IF(ITIME(1).EQ.2HIN)GOTO100
1036      IF(CKTM(IBTIM,ISTM))100,300,250
1037 C      IF NOT THERE YET, GO BACK TO 100 AND CONTINUE
1038 C      IF EXACTLY THERE, GOTO 300 AND RETURN
1039 C      IF TIME NOW IS GREATER THAN REQUESTED, ADJUST TIME AND RETURN
1040 250   BACKSPACE(LUIN)
1041      NTAP=5
1042      CALL REDAT(IEOF,1)
1043      DO 260 I=1,6
1044 260   ISTM(I)=IBTIM(I)
1045 300   IERR=0
1046 900   RETURN
1047      END

```

FTH4X COMPILER: HP92834 REV.2130 (810716)

\*\* NO WARNINGS \*\* NO ERRORS \*\* PROGRAM: 231 COMMON: 1526

```

1048 SUBROUTINE REDAT(IEOF,ITYP),READ NEXT ITP RECORD
1049 C
1050 C
1051 C REDAT READS MAJOR FRAMES OF HALOE DATA FROM THE INPUT FILE.
1052 C IEOF IS A FLAG PASSED BACK TO MAIN PROGRAM INDICATING END-OF-FILE
1053 C STATUS (= TRUE IF EOF)
1054 C
1055 COMMON/IDAT/IBUF(256),IFLAG,IBTIM(6),ISTAT(10),IANHK(24),IPWR(4)
1056 *,ITYPE
1057 COMMON LUT,LULOG,LUIN,LUWIN,N,INBUF(10),LBUF(1510),LUPR
1058 C
1059 C N= 5 WHEN PROCESSING MAG TAPE FIRST TIME, OR WHEN SEEKING NEW
1060 C TIME ON MAG TAPE. NOT USED IN DISK FILE MANIPULATION.
1061 C
1062 DIMENSION IBUFF(1510)
1063 LOGICAL IEOF
1064 IF(LUIN.EQ.40)THEN
1065 1 READ(LUIN,END=900,ERR=6 ,IOSTAT=IOS)ITYPE,
1066 *IPWR,IBTIM,IBUF,IDUM,IANHK,ISTAT
1067 C WRITE(LULOG,1001)ITYPE,IBTIM
1068 C1001 FORMAT(" RECORD TYPE, TIME ", 7I5)
1069 IF(ITYP.EQ.0)GOTO5
1070 IF(ITYPE.NE.ITYP)GOTO1
1071 5 IEOF=.FALSE.
1072 RETURN
1073 6 WRITE(LULOG,1002)
1074 1002 FORMAT(" END OF FILE ENCOUNTERED,REWIND,CONTINUE OR STOP?"
1075 *" (R/C/S)")
1076 READ(LUT,1003)IANS
1077 1003 FORMAT(A1)
1078 IF(IANS.EQ.1HC)GOTO1
1079 IF(IANS.EQ.1HR)THEN
1080 REWIND(LUIN)
1081 GOTO1
1082 ENDIF
1083 GOTO900
1084 100 IF(100.EQ.496)GOTO1 ! ERROR WAS DUE TO SMALLER RECORD TYPE
1085 WRITE(LULOG,1000)IOS
1086 1000 FORMAT(" ERROR # ",15," IN REDAT ROUTINE")
1087 STOP
1088 ELSE
1089 3 N=N+1
1090 IF(N.GE.5)THEN
1091 N=0
1092 2 READ(LUIN,END= 7 ,ERR=1800,IOSTAT=IOS)IBUFF
1093 ENDF
1094 NN=N*302
1095 ITYPE=IBUFF(NN+1)
1096 IF(ITYP.EQ.0)GOTO4
1097 IF(ITYPE.NE.ITYP)GOTO3
1098 4 IEOF=.FALSE.
1099 CALL MVARY(IBUFF(NN+6),IBTIM(1),6)
1100 CALL MVARY(IBUFF(NN+12),IBUF(1),256)
1101 CALL MVARY(IBUFF(NN+293),ISTAT(1),10)
1102 CALL MVARY(IBUFF(NN+2),IPWR(1),4)

```

```
1103      CALL MVARY(IBUFF(NN+269),IANHK(1),24)
1104 C      WRITE(LULOG,1001)ITYPE,IBTIM
1105      RETURN
1106 7      WRITE(LULOG,1002)
1107      READ(LUT,1003)IANS
1108      IF(IANS.EQ.1HC)GOTO2
1109      IF(IANS.EQ.1HR)THEN
1110          REWIND(LUIN)
1111          GOTO2
1112      ENDIF
1113 900      IEOF=.TRUE.
1114      RETURN
1115 1800     IF(IOS.EQ.496)GOTO2
1116      ENDIF
1117      RETURN
1118      END
```

FTN4X COMPILER: HP92834 REV.2130 (810716)

\*\* NO WARNINGS \*\* NO ERRORS \*\* PROGRAM: 1828 COMMON: 1526



## APPENDIX B - HPLOT

Program Name: HPLOT

Function: HPLOT is used to plot HALOE Blackbody data.

Description: HPLOT is a Fortran V program written on the ACD NOS facility.

Use: HPLOT can be executed using the procedure listed below. The plots will be routed to the Calcomp plotters automatically.

```
.PROC,HPLOTPR,TAPEND.
```

```
GET,HPLOT.
```

```
FTNS,I=HPLOT,L=LF.
```

```
ATTACH,LARCGOS/UN=LIBRARY,NA.
```

```
COMMENT.PROCESSING DONE FOR TAPEND DATA.
```

```
GET,TAPE1=TAPEND.
```

```
LDSET,LIB=LARCGOS,PRESSTA=NGINF.
```

```
LGO.
```

```
.NOTE,(/IF YOU WANT A PRINTED OUTPUT OF DAILY AND WEEKLY
```

```
.NOTE,AVERAGE ROUTE THE TAPE4 TO LINE PRINTER AS FOLLOWS
```

```
.NOTE,ROUTE,TAPE4,DC=LF/)
```

```
REVERT.
```

```

PROGRAM HPLOT
74/R60 OPT=1,ROUND= A/ S/ M/-D,-DS      FTN 5.1+642      87/04/30.
DO=-LONG/-OT,ARG= COMMON/--FIXED,CS= USER/--FIXED,DB=-TB/-SB/-SL/-ER/-ID/-PMD/-ST,-AL,PL=5000
FTN5,I=HPLOT,L=LF.

```

```

1  PROGRAM HPL0T
2
3  *THIS SOFTWARE TESTS THE PERFORMANCE OF
4  *THE BLACKBODY OF THE HALOE INSTRUMENT.
5  *THE MAIN OBJECTIVE FOR DEVELOPING THIS PROGRAM IS TO
6  *FIND THE CO-RELATION BETWEEN PRT(PLATINUM RESISTANCE
7  *THERMOMETER) AND OTHER RELATED MEASUREMENTS SUCH AS
8  *CURRENT, POWER SUPPLY VOLTAGE, ENVIRONMENTAL TEMPERATURE
9  *CHANGESSUCH AS VACUUM CHAMBER WALL TEMPERATURE,BLACKBODY
10 *CASE TEMPERATURE,BB ISOLATOR MOUNT TEMPERATURE..ETC.
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40

```

```

41 * 2.PLOT PRT WITH BBP AND BBR USING BBH *
42 * 3.PLOT PRT WITH TCKT AND TCHBR *
43 * 4.PLOT PRT WITH TISOL BBP AND TCASE *
44 * 5.PLOT PRT WITH VPS AND R *
45 * 6.PLOT BBR BBR WITH TCKT AND TCHBR *
46 * 7.PLOT PRT WITH PRESSURE *
47 * 8.GET THE DAILY AVERAGE OF ALL THE PARAMETERS **
48 * 9.QUIT. *
49 *
50 *****
51 *****
52 *****
53 *****
54 *****
55 *****
56 *****
57 *****
58 *****
59 *****
60 *****
61 *****
62 *****
63 *****
64 *****
65 *****
66 *****
67 *****
68 *****
69 *****
70 *****
71 *****
72 *****
73 *****
74 *****
75 *****
76 *****
77 *****
78 *****
79 *****
80 *****
81 *****
82 *****

```

```

83 39 READ *,ICAP
84 IF(ICAP.EQ.10)GO TO 999
85 PRINT *,ENTER THE YEAR (YYYY),
86 READ *,IYEAR
87 PRINT *,ENTER THE STARTING DAY (DDD),
88 READ *,IDAYS
89 PRINT *,ENTER THE STARTING TIME (HH,MM),
90 READ *,IHR,IMIN
91 PRINT *,ENTER THE ENDING DAY (DDD),
92 READ *,IDAYE
93 PRINT *,ENTER THE ENDING TIME (HH,MM),
94 READ *,IHRE,IMINE
95
96 C*****THE CALIBRATION FACTOR FOR COMPUTING PRT IS .0954
97 C*****THE MULTIPLIER FOR THE CURRENT IS 2
98 C*****THE CALIBRATION FACTOR FOR THE VACCUUM SYSTEM IS 10-7
99 C
100 PRINT *,ENTER THE CALIBRATION FACTOR TO COMPUTE PRT,
101 READ *,CALFAC
102 PRINT *,ENTER MULTIPLIER FOR CURRENT,
103 READ *,CURMUL
104 PRINT *,ENTER CALIBRATION FOR THE VACUUM SYSTEM,
105 READ *,CALVAC
106
107 *CONVERT THE STARTING TIME TO HOURS
108
109 C
110 STIME=FLOAT(IHR)+FLOAT(IMIN)/60.
111
112 *CONVERT THE ENDING TIME TO HOURS
113 C
114 ETIME=FLOAT(IHRE)+FLOAT(IMINE)/60.+((IDAYE-IDAYS)*24)
115 RHR=STIME
116 EHR=ETIME
117 WRITE(3,51)ETIME
118 FORMAT(2X,'ETIME',2X,F8.4)
119 REWIND 1
120
121 C
122 C*****READ THE DATA FILE
123 C
124 C

```

87/04/30. 09.48.

FTN 5.1+642

74/860 OPT=1,ROUND= A/ S/ M/-D,-DS

PROGRAM HPLDT

```

125 READ(1,900,END=18)NDAY,NHR,NMIN,BBV,BBI,TCKT,TISOL,
126 *TCHBR,PRT,TCASE,VREF,PRESS,RADIO,VPS,BBH
127 FORMAT(3I4,12F8.3)
128 WRITE(3,47)NDAY,NHR,NMIN
129 FORMAT(2X,'DAY',2X,3(I6))
130 C
131 C
132 C
133 DTIME=FLOAT(NHR)+FLOAT(NMIN)/60.+FLOAT((NDAY-IDAYS)*24)
134 WRITE(3,52)DTIME
135 FORMAT(2X,'DTIME',2X,F8.4)
136 C
137 C
138 C
139 *IF THE ENDING TIME IS REACHED GO TO STATEMENT NO.19
140 IF(DTIME.GT.ETIME)THEN
141 GO TO 19
142 ENDIF
143 IF(NDAY.LT.IDAYS)THEN
144 GO TO 6
145 ELSEIF(NDAY.GT.IDAYS .AND. NUM.EQ.0)THEN
146 PRINT *,IDAYS,NDAY
147 FORMAT(2X,I6,2X,I6)
148 PRINT 100,NDAY,NHR,NMIN
149 PRINT *,'DO YOU WANT TO ENTER THE DAY AND TIME AGAIN?Y/N'
150 READ (*,'(A1)')ANS
151 IF (ANS.EQ. 'Y')THEN
152 GO TO 10
153 ELSEIF(ANS.EQ.'N')THEN
154 GO TO 1000
155 ENDIF
156 PRINT *,'ERROR IN INPUT '
157 GO TO 5
158 ENDIF
159 DTIME=FLOAT(NHR)+FLOAT(NMIN)/60.+FLOAT((NDAY-IDAYS)*24)
160 IF(DTIME.GT. ETIME .AND. NUM .EQ. 0)THEN
161 PRINT *,'STARTING TIME NOT IN THE TAPE '
162 GO TO 1000
163 C
164 C
165 C
166 C
167 ENDIF
168 IF(DTIME.LT.ETIME)THEN
169 GO TO 6
170 ELSEIF(DTIME.LE.ETIME)THEN
171 IF(NUM.GT.5000)THEN

```

```
167 PRINT *, 'DATA POINTS MORE THAN 5000'
168 PRINT *, 'PLEASE START ALL OVER AGAIN'
169 GO TO 1000
170
171 ELSE
172
173 C
174 C
175 C
176
177 *START ACCUMULATING THE DATA POINTS TO PLOT
178
179 NUM=NUM+1
180 JDAY(NUM)=NDAY
181 X(NUM)=OTIME
182
183 WRITE(3,28)X(NUM)
184 FORMAT(2X,'TIME',2X,F7.2)
185 Y1(NUM)=1000.0 - PRT / CALFAC
186 WRITE(3,12)Y1(NUM)
187 FORMAT(2X,'PRT',2X,F7.2)
188 Y2(NUM)=BBV
189 BBI=BBI * CURMUL
190 WRITE(3,14)BBI
191 FORMAT(2X,'BBI',2X,F7.2)
192 Y3(NUM)=BBI
193 Y4(NUM)=TCKT
194 Y5(NUM)=TISOL
195 Y6(NUM)=TCHBR
196 Y7(NUM)=TCASE
197 Y8(NUM)=RADIO
198 Y9(NUM)=BBH
199 Y10(NUM)=VPS
200 Y11(NUM)=BBV*BBI
201 WRITE(3,16)Y11(NUM)
202 FORMAT(2X,'BBP',2X,F7.2)
203 Y12(NUM)=BBV/BBI
204 WRITE(3,17)Y12(NUM)
205 FORMAT(2X,'BBR',2X,F7.2)
206 Y14(NUM)=BBH*BBI
207 WRITE(3,25)Y14(NUM)
208 FORMAT(2X,'BBP USING BBH',2X,F7.2)
209 Y15(NUM)=BBH/BBI
210 WRITE(3,26)Y15(NUM)
211 FORMAT(2X,'BBR USING BBH',2X,F7.2)
212 Y16(NUM)=PRESS*CALVAC
213 GO TO 6
214 ENDIF
```

1 209  
1 210  
1 211  
1 212  
1 213  
1 214  
1 215  
1 216  
1 217  
1 218  
1 219  
1 220  
1 221  
1 222  
1 223  
1 224  
1 225  
1 226  
1 227  
1 228  
1 229  
1 230  
1 231  
1 232  
1 233  
1 234  
1 235  
1 236  
1 237  
1 238  
1 239  
1 240  
1 241  
1 242  
1 243  
1 244  
1 245  
1 246  
1 247  
1 248  
1 249  
1 250

ENDIF  
\*GET THE DAILY AVERAGE

C  
19

SPRT=0.  
SBBP=0.  
SBBR=0.  
SCKT=0.  
SISOL=0.  
SCHBR=0.  
SCASE=0.  
SRAD=0.  
SPRESS=0.  
SVPS=0.

C  
C

KDAY=JDAY(1)

M=0

NPTS=0

DO 575 I=1,NUM

C  
C

IF (JDAY(I) .GT. KDAY .OR. I .EQ. NUM) THEN

M=M+1

DPRT(M) = SPRT / NPTS

DBBP(M) = SBBP / NPTS

DBBR(M) = SBBR / NPTS

DCKT(M) = SCKT / NPTS

DISOL(M) = SISOL / NPTS

DCHBR(M) = SCHBR / NPTS

DCASE(M) = SCASE / NPTS

DRAD(M) = SRAD / NPTS

DVPS(M) = SVPS / NPTS

DPRESS(M) = SPRESS / NPTS

DDAY(M) = FLOAT(KDAY)

WRITE(4,27)DDAY(M),DPRT(M),DBBP(M),DBBR(M),DCKT(M),DISOL(M),

DCHBR(M),DCASE(M),DRAD(M),DVPS(M),DPRESS(M)

FORMAT(11(1X,F9.3))

27

C  
C

NPTS=1

SPRT=Y1(I)

SBBP=Y11(I)

SBBR=Y12(I)

ORIGINAL PAGE IS  
OF POOR QUALITY

1	251
1	252
1	253
1	254
1	255
1	256
1	257
1	258
1	259
1	260
1	261
1	262
1	263
1	264
1	265
1	266
1	267
1	268
1	269
1	270
1	271
1	272
	273
	274
	275
	276
	277
	278
	279
	280
	281
	282
	283
	284
	285
	286
	287
	288
	289
	290
	291
	292

**B-8**

```

SCKT=Y4(I)
SISOL=Y5(I)
SCHBR=Y6(I)
SCASE=Y7(I)
SRAD=Y8(I)
SPRESS=Y16(I)
SVPS=Y10(I)
KDAY=JDAY(I)
ELSE
  SPRT = SPRT + Y1(I)
  SBBP = SBBP + Y11(I)
  SBBR = SBBR + Y12(I)
  SCKT = SCKT + Y4(I)
  SISOL = SISOL + Y5(I)
  SCHBR = SCHBR + Y6(I)
  SCASE = SCASE + Y7(I)
  SRAD = SRAD + Y8(I)
  SPRESS = SPRESS + Y16(I)
  SVPS = SVPS + Y10(I)
  NPTS = NPTS + 1
ENDIF
CONTINUE
*GET THE WEEKLY AVERAGE

WRITE(4,143)
FORMAT(2X,'WEEKLY AVERAGES')
DO 1050 IN=1,15
  WPRT(IN)=0.
  WBBP(IN)=0.
  WBBR(IN)=0.
  WCKT(IN)=0.
  WISOL(IN)=0.
  WCHBR(IN)=0.
  WCASE(IN)=0.
  WRAD(IN)=0.
  WVPS(IN)=0.
  WPRESS(IN)=0.
CONTINUE
NW=M/7
DO 590 LW=1,NW
  ID=(LW-1)*7 + 1

```

ORIGINAL PAGE IS  
OF POOR QUALITY



```

293 IDE=ID+6
294 DO 580 KD=ID,IDE
295   WPRT(LW)=WPRT(LW)+DPRT(KD)
296   WBBP(LW)=WBBP(LW)+D8BP(KD)
297   WBBR(LW)=WBBR(LW)+D8BR(KD)
298   WCKT(LW)=WCKT(LW)+DCKT(KD)
299   WISOL(LW)=WISOL(LW)+DISOL(KD)
300   WCHBR(LW)=WCHBR(LW)+DCHBR(KD)
301   WCASE(LW)=WCASE(LW)+DCASE(KD)
302   WRAD(LW)=WRAD(LW)+DRAD(KD)
303   WVPS(LW)=WVPS(LW)+DVPS(KD)
304   WPRESS(LW)=WPRESS(LW)+DPRESS(KD)
305   CONTINUE
306   WPRT(LW)=WPRT(LW)/7.
307   WBBP(LW)=WBBP(LW)/7.
308   WBBR(LW)=WBBR(LW)/7.
309   WCKT(LW)=WCKT(LW)/7.
310   WISOL(LW)=WISOL(LW)/7.
311   WCHBR(LW)=WCHBR(LW)/7.
312   WCASE(LW)=WCASE(LW)/7.
313   WRAD(LW)=WRAD(LW)/7.
314   WVPS(LW)=WVPS(LW)/7.
315   WPRESS(LW)=WPRESS(LW)/7.
316   WWK(LW)=LW
317   WRITE(4,165)WWK(LW),WPRT(LW),WBBP(LW),WBBR(LW),WCKT(LW),
318   *WISOL(LW),WCHBR(LW),WCASE(LW),WRAD(LW),WPRESS(LW)
319   FORMAT(11(1X,F9.3))
320   CONTINUE
321   CONTINUE
322   WRITE(3,30)DPRT(1),D8BP(1),D8BR(1),M
323   FORMAT(1X,3F7.1,1X,I3)
324   FORMAT(2X,'YOU HAVE ASKED FOR DAY BEFORE THE DATA PERIOD'/
325   *'THE STARTING TIME ON THE TAPE IS ',2X,I4,2X,I2,2X,I2)
326   DO 400 I=2,NUM
327     X(I)=X(I)-X(1)
328     CONTINUE
329     X(1)=0.
330     FYEAR=FLOAT(IYEAR)
331     FHR=FLOAT(IHR)
332     FMIN=FLOAT(IMIN)
333     FHRE=FLOAT(IHRE)
334     FMINE=FLOAT(IMINE)

```

ORIGINAL PAGE IS  
OF POOR QUALITY

```
335 BDAY=JDAY(1)
336 SDAY=JDAY(NUM)
337
338
339 *SUBROUTINE OMDATE IS CALLED TO CONVERT THE DAY NUMBER
340 TO MONTH AND DATE
341
342
343 CALL OMDATE(BDAY,SDAY)
344 XSCALE=(EHR - BHR) / 8.
345
346
347 *BRANCHES TO RESPECTIVE SUBROUTINES DEPENDING UPON THE
348 *SELECTION OF THE OPTION
349
350
351 IF(ICAP.EQ.1)THEN
352   CALL SUB1(Y1,Y11,Y12,Y14,Y15,X)
353   ELSEIF(ICAP.EQ.2)THEN
354     CALL SUB2(Y1,Y14,Y15,X)
355     ELSEIF(ICAP.EQ.3)THEN
356       CALL SUB3(Y1,Y4,Y6,X,Y11,Y12)
357       ELSEIF(ICAP.EQ.4)THEN
358         CALL SUB4(Y1,Y5,Y7,Y11,X)
359         ELSEIF(ICAP.EQ.5)THEN
360           CALL SUB5(Y1,Y10,Y8,X)
361           ELSEIF(ICAP.EQ.6)THEN
362             CALL SUB6(Y11,Y12,Y4,Y6,X)
363             ELSEIF(ICAP.EQ.7)THEN
364               CALL SUB7(Y1,Y16,X)
365               ELSEIF(ICAP.EQ.8)THEN
366                 CALL SUB8(DDAY,DPRT,DBBP,DBBR,DCKT,DISOL,
367                   DCHBR,DCASE,DRAD,DVPS,DPRESS,M,IDAYS,IDAYE)
368                 ELSEIF(ICAP.EQ.9)THEN
369                   CALL SUB9(WWK,WPRT,WBBP,WBBR,WCKT,WISOL,
370                     WCHBR,WCASE,WRAD,WVPS,WPRESS,NW)
371                   ELSEIF(ICAP.EQ.10)THEN
372                     GO TO 999
373
374   ENDIF
375   REWIND 1
376   GO TO 1
```



```
1 C
2 C
3 C
4 C
5 C
6
7 SUBROUTINE DMDATE(BDAY,SDAY)
8 COMMON /DATE/FMNTNTH,FDAY,EMNTH,EDAY
9 INTEGER BDAY,SDAY
10 DIMENSION MNTH(12)
11
12 DATA MNTH /1,32,60,91,121,152,182,213,244,274,305,335/
13
14 DO 500 I=1,11
15 IF(BDAY.LT.MNTH(I+1))THEN
16 FMNTH=FLOAT(I)
17 FDAY=FLOAT((BDAY-MNTH(I))+1)
18 GO TO 600
19 ENDIF
20
21 CONTINUE
22
23 FMNTH=12
24 FDAY=FLOAT((BDAY-MNTH(12))+1)
25
26 CONTINUE
27
28 DO 800 I=1,11
29 IF(SDAY.LT.MNTH(I+1))THEN
30 EMNTH=FLOAT(I)
31 EDAY=FLOAT((SDAY-MNTH(I))+1)
32 GO TO 900
33 ENDIF
34 CONTINUE
35
36 EMNTH=12
37 EDAY=FLOAT((SDAY-MNTH(12))+1)
38
39 CONTINUE
40
```

41 RETURN  
42 END

--VARIABLE MAP--(LO=A)  
--NAME--ADDRESS --BLOCK--PROPERTIES--SIZE --TYPE--TYPE--ADDRESS --BLOCK--PROPERTIES  
B DAY 1 DUMMY-ARG INTEGER FMNTH OB /DATE/  
E DAY 3B /DATE/ I 65B  
E MNTH 2B /DATE/ MNTH 51B  
F DAY 1B /DATE/ SDAY 2 DUMMY-ARG

--PROCEDURES--(LO=A)  
--NAME--TYPE--ARGS--CLASS--  
FLOAT REAL 1 INTRINSIC

--STATEMENT LABELS--(LO=A)  
--LABEL-ADDRESS--PROPERTIES--DEF --LABEL-ADDRESS--PROPERTIES--DEF  
500 INACTIVE DO-TERM 22 800 INACTIVE DO-TERM 35  
550 \*NO REFS\* 24 850 \*NO REFS\* 37  
600 25B 27 900 44B 40

--ENTRY POINTS--(LO=A)  
--NAME--ADDRESS--ARGS--  
OMDATE 3B 2

--STATISTICS--  
PROGRAM-UNIT LENGTH 72B = 58  
CM LABELLED COMMON LENGTH 48 = 4  
CM STORAGE USED 61400B = 25344  
COMPILE TIME 0.384 SECONDS

ORIGINAL PAGE IS  
OF POOR QUALITY

1

1111111111

ORIGINAL PAGE IS  
OF POOR QUALITY

```
1 41
1 42
1 43
44 PRINT *, 'WRONG INPUT'
45 GO TO 5
100
46 ENDIF
47 Y1(NUM+1)=PRTMIN
48 Y1(NUM+2)=PRTSF
49 Y11(NUM+1)=BBPMIN
50 Y11(NUM+2)=BBPSF
51 Y12(NUM+1)=BBRMIN
52 Y12(NUM+2)=BBRSF
53 CALL LEPOY
54 CALL NEWPEN(1)
55 CALL CALPLT(2,1,-3)
56 X(NUM+1)=0.
57 X(NUM+2)=XSCALE
58 CALL AXES(0,0,0,0,9,X(NUM+1),X(NUM+2),1,10,ISTRIN,14,-18)
59 CALL AXES(0,0,0,90,5,Y1(NUM+1),Y1(NUM+2),1,10,'PRT',14,3)
60 CALL AXES(0,5,0,9,X(NUM+1),X(NUM+2),1,10,'',0,0,1)
61 CALL AXES(9,0,90,5,Y1(NUM+1),Y1(NUM+2),1,10,'',0,0,-1)
62 CALL NEWPEN(1)
63 CALL LINPLT(X,Y1,NUM,1,0,0,1,1)
64 CALL NEWPEN(2)
65 CALL CALPLT(-1,0,-3)
66 CALL AXES(0,0,90,5,Y11(NUM+1),Y11(NUM+2),1,10,'BBP',
67 *14,3)
68 CALL CALPLT(1,0,0,-3)
69 CALL NEWPEN(2)
70 CALL LINPLT(X,Y11,NUM,1,0,0,1,2)
71 CALL NEWPEN(3)
72 CALL CALPLT(10,0,-3)
73 CALL AXES(0,0,90,5,Y12(NUM+1),Y12(NUM+2),1,10,'BBR',
74 *14,3)
75 CALL CALPLT(-10,0,0,-3)
76 CALL NEWPEN(3)
77 CALL LINPLT(X,Y12,NUM,1,0,0,1,3)
78 CALL NEWPEN(1)
79 CALL HEADR
80 CALL NFRAME
81 RETURN
82 END
```

SUBROUTINE SUB2  
74/860 OPT=1,ROUND= A/ S/ M/-D,-DS FTN 5.1+642 87/04/30. 09.48.C  
DO=-LONG/-OT,ARG= COMMON/-FIXED,CS= USER/-FIXED,DB=-TB/-SB/-SL/-ER/-ID/-PMD/-ST,-AL,PL=5000  
FTN5,I=HPLOT,L=LF.

```

1 SUBROUTINE SUB2(Y1,Y14,Y15,X)
2
3 C*****THE FOLLOWING SUBROUTINE SUB2 PLOTS PRT,BBP,BBR**
4 C*****THIS USES BBH TO COMPUTE BBPAND BBR***
5
6 COMMON/TOP/ANS,ISTRIN
7 COMMON/SCALE/XSCALE
8 COMMON/TIM/FYEAR,FHR,FMIN,FHRE,FMINE
9 COMMON/DATE/FMINTH,FDAY,EMNTH,EDAY
10 COMMON/KEY/ PRTMIN,PRTSF,BBPMIN,BBPSF,BBRMIN,BBRSF
11 COMMON/KEY1/ TCKMIN,TCKSF,TCHMIN,TCHSF,TSOLMIN,TSOLSF,NUM,
12 *TCASMIN,TCASSF,VPSSMIN,VPSSF,RADMIN,RADSF,PREMIN,PRESF
13 CHARACTER ANS*1,ISTRIN*18
14
15 DIMENSION Y1(NUM+2),Y14(NUM+2),Y15(NUM+2),X(NUM+2)
16 DATA ISTRIN/'ELAPSED TIME(HRS)'/
17 PRINT *, 'THE DEFAULT GRAPHING LIMITS ARE AS FOLLOWS : '
18 PRINT *, 'PRT MIN .....980K'
19 PRINT *, 'PRT SF .....10'
20 PRINT *, 'BBP MIN .....2WATTS'
21 PRINT *, 'BBP SF .....2'
22 PRINT *, 'BBR MIN .....3.5 OHMS'
23 PRINT *, 'BBR SF .....(.5)'
24 PRINT *, 'DO YOU WANT TO CHANGE THE DEFAULT MIN AND SF?(Y/N)'
25 READ (*, '(A1)')ANS
26 IF(ANS.EQ.'Y')THEN
27 PRINT *, 'INPUT THE NEW PRT MIN AND SF'
28 READ *, PRTMIN,PRTSF
29 PRINT *, 'INPUT THE NEW BBP MIN AND SF'
30 READ *, BBPMIN,BBPSF
31 PRINT *, 'INPUT THE NEW BBR MIN AND SF'
32 READ *, BBRMIN,BBRSF
33 ELSEIF(ANS.EQ.'N')THEN
34 GO TO 100
35 ELSEIF(ANS.NE.'Y'.OR.ANS.NE.'N')THEN
36 PRINT *, 'WRONG INPUT'
37 GO TO 5
38
39 ENDIF
40 Y1(NUM+1)=PRTMIN
41 Y1(NUM+2)=PRTSF

```



```
41 Y14(NUM+1)=BBPMIN
42 Y14(NUM+2)=BBPSF
43 Y15(NUM+1)=BBRMIN
44 Y15(NUM+2)=BBRSF
45 CALL LERDY
46 CALL CALPLT(2,,1,-3)
47 X(NUM+1)=0.
48 X(NUM+2)=XSCALE
49 CALL AXES(0,,0,,0,,9,,X(NUM+1),X(NUM+2),1,,10,,ISTRIN,.14,-18)
50 CALL AXES(0,,0,,90,,5,,Y1(NUM+1),Y1(NUM+2),1,,10,,PRT,.14,3)
51 CALL AXES(0,,5,,0,,9,,X(NUM+1),X(NUM+2),1,,10,,.1,0,0,1)
52 CALL AXES(9,,0,,90,,5,,Y1(NUM+1),Y1(NUM+2),1,,10,,.1,0,0,-1)
53 CALL NEWPEN (1)
54 CALL LINPLT(X,Y1,NUM,1,0,0,1,1)
55 CALL NEWPEN (2)
56 CALL CALPLT(-1,,0,-3)
57 CALL AXES(0,,0,,90,,5,,Y14(NUM+1),Y14(NUM+2),1,,10,,'BBP',
58 *14,3)
59 CALL CALPLT(1,,0,-3)
60 CALL LINPLT(X,Y14,NUM,1,0,0,1,2)
61 CALL NEWPEN (3)
62 CALL CALPLT(10,,0,-3)
63 CALL AXES(0,,0,,90,,5,,Y15(NUM+1),Y15(NUM+2),1,,10,,'BBR',
64 *14,3)
65 CALL CALPLT(-10,,0,-3)
66 CALL LINPLT(X,Y15,NUM,1,0,0,1,3)
67 CALL NEWPEN(1)
68 CALL HEADR
69 CALL NFRAME
70 RETURN
71 END
```

ORIGINAL PAGE IS  
OF POOR QUALITY

---VARIABLE MAP---(LO=A)

---NAME---	---ADDRESS---	---BLOCK---	---PROPERTIES---	---TYPE---	---SIZE---	---NAME---	---ADDRESS---	---BLOCK---	---PROPERTIES---
ANS	0B	/TOP/		CHAR*1		EMNTH	2B	/DATE/	
BBPMIN	2B	/KEY/	REAL	REAL		FDAY	1B	/DATE/	
BBPSF	3B	/KEY/	REAL	REAL		FHR	1B	/TIM/	
BBRMIN	4B	/KEY/	REAL	REAL		FHRE	3B	/TIM/	
BBRSF	5B	/KEY/	REAL	REAL		FMIN	2B	/TIM/	
EDAY	3B	/DATE/	REAL	REAL		FMINE	4B	/TIM/	

```

1  SUBROUTINE SUB3(Y1,Y4,Y6,X,Y11,Y12)
2
3  C
4  C
5  C*****THE FOLLOWING SUBROUTINE SUB3 PLOTS PRT,CIRCUIT TEMP*
6  C*****AND BB CHAMBER TEMP*****
7  C
8  COMMON/TOP/ANS,ISTRIN
9  COMMON/SCALE/XSCALE
10 COMMON/TIM/FYEAR,FHR,FMIN,FHRE,FHRE,FHRE,FHRE
11 COMMON/DATE/FMINTH,FDAY,EMINTH,EDAY
12 COMMON/KEY/ PRTMIN,PRTSF,BBPMIN,BBPSF,BBPMIN,BBPSF
13 COMMON/KEY1/ TCKMIN,TCKSF,TCHMIN,TCHSF,TCKMIN,TCKSF,TCHMIN,TCKSF,TCHMIN,TCKSF
14 *TCASMIN,TCASSF,VPSMIN,VPSSF,RADMIN,RADSF,PREMIN,PRESF
15 DIMENSION Y1(NUM+2),Y4(NUM+2),Y6(NUM+2),X(NUM+2),Y11(NUM+2),
16 *Y12(NUM+2)
17 CHARACTER ANS*1,ISTRIN*18
18 DATA ISTRIN/'ELAPSED TIME(HRS)'/
19 PRINT *, 'THE DEFAULT GRAPHING LIMITS ARE AS FOLLOWS : '
20 PRINT *, 'PRT MIN .....980 K'
21 PRINT *, 'PRT SF .....10'
22 PRINT *, 'TCKT MIN .....20 DEG'
23 PRINT *, 'TCKT SF .....5'
24 PRINT *, 'TCH MIN .....15 DEG'
25 PRINT *, 'TCH SF .....(5)'
26 PRINT *, 'BBP MIN .....2 WATTS'
27 PRINT *, 'BBP SF .....2'
28 PRINT *, 'BBR MIN .....2.5 OHMS'
29 PRINT *, 'BBR SF .....(5)'
30 PRINT *, 'DO YOU WANT TO CHANGE THE DEFAULT MIN AND SF ?(Y/N)'
31 READ (*, '(A1)')ANS
32 IF(ANS.EQ. 'Y')THEN
33 PRINT *, 'INPUT THE NEW MIN AND SF PRT VALUES'
34 READ *, PRTMIN, PRTSF
35 PRINT *, 'INPUT THE NEW MIN AND SF T CKT VALUES'
36 READ *, TCKMIN, TCKSF
37 PRINT *, 'INPUT THE NEW MIN AND SF T CHBR VALUES'
38 READ *, TCHMIN, TCHSF
39 PRINT *, 'INPUT THE NEW BBP MIN AND SF'
40 READ *, BBPMIN, BBPSF

```

ORIGINAL PAGE IS  
OF POOR QUALITY

```

1 41 PRINT *, 'INPUT THE NEW BBR MIN AND SF'
1 42 READ *, BBRMIN, BBRSE
1 43 ELSEIF (ANS .EQ. 'N') THEN
1 44 GO TO 100
1 45 ELSEIF (ANS .NE. 'Y' .OR. ANS .NE. 'N') THEN
1 46 PRINT *, 'WRONG INPUT'
1 47 GO TO 5
1 48
1 49
1 50 100 Y1(NUM+1)=PRTMIN
1 51 Y1(NUM+2)=PRTSF
1 52 Y4(NUM+1)=TCKMIN
1 53 Y4(NUM+2)=TCKSF
1 54 Y6(NUM+1)=TCHMIN
1 55 Y6(NUM+2)=TCHSF
1 56 CALL LEROY
1 57 CALL CALPLT(2,1,-3)
1 58 X(NUM+1)=0.
1 59 X(NUM+2)=XSCALE
1 60 CALL AXES(0,0,0,0,9,X(NUM+1),X(NUM+2),1,10,ISTRIN,.14,-18)
1 61 CALL AXES(0,0,0,90,5,Y1(NUM+1),Y1(NUM+2),1,10,'PRT',.14,3)
1 62 CALL AXES(0,5,0,9,X(NUM+1),X(NUM+2),1,10,'',0,0,1)
1 63 CALL AXES(9,0,90,5,Y1(NUM+1),Y1(NUM+2),1,10,'',0,0,-1)
1 64 CALL NEWPEN(1)
1 65 CALL LINPLT(X,Y1,NUM,1,0,0,1,1)
1 66 CALL NEWPEN(2)
1 67 CALL CALPLT(-1,0,-3)
1 68 CALL AXES(0,0,90,5,Y4(NUM+1),Y4(NUM+2),1,10,'CIRCUIT',
1 69 *.14,7)
1 70 CALL CALPLT(1,0,-3)
1 71 CALL LINPLT(X,Y4,NUM,1,0,0,1,2)
1 72 CALL NEWPEN(3)
1 73 CALL CALPLT(10,0,-3)
1 74 CALL AXES(0,0,90,5,Y6(NUM+1),Y6(NUM+2),1,10,'CHAMBER',
1 75 *.14,7)
1 76 CALL CALPLT(-10,0,-3)
1 77 CALL LINPLT(X,Y6,NUM,1,0,0,1,3)
1 78 CALL NEWPEN(1)
1 79 CALL HEADR
1 80 CALL NFRAME
1 81 RETURN
1 82 END

```

ORIGINAL PAGE IS  
 OF POOR QUALITY

```

1  C
2  SUBROUTINE SUB4(Y1,Y5,Y7,Y11,X)
3  C
4  C
5  *THE FOLLOWING SUBROUTINE SUB4 PLOTS PRT,ISOLATOR TEMP**
6  *BB CASE TEMP BBP*****
7  C
8  COMMON/TOP/ANS,ISTRIN
9  COMMON/SCALE/XSCALE
10 COMMON/TIM/FYEAR,FHR,FMIN,FHRE,FHNE
11 COMMON/DATE/FMINTH,FDAY,EMINTH,EDAY
12 COMMON/KEY/ PRTMIN,PRTSF,BBPMIN,BBPSF,BBRMIN,BBRSF
13 COMMON/KEY1/ TCKMIN,TCKSF,TCHMIN,TCHSF,TSOLMIN,TSOLSF,NUM,
14 *TCASMIN,TCASSF,VPSMIN,VPSSF,RADMIN,RADSF,PREMIN,PRESF
15 CHARACTER ANS*1,ISTRIN*18
16 DIMENSION Y1(NUM+2),Y5(NUM+2),Y7(NUM+2),Y11(NUM+2),X(NUM+2)
17 DATA ISTRIN/'ELAPSED TIME(HRS)'/
18 PRINT *, 'THE DEFAULT GRAPHING LIMITS ARE AS FOLLOWS : '
19 PRINT *, 'PRTMIN .....980 K'
20 PRINT *, 'PRTSF .....10'
21 PRINT *, 'T ISOL MIN ...110 DEG'
22 PRINT *, 'T ISOL SF ... (5.)'
23 PRINT *, 'T CASE MIN ...115 DEG'
24 PRINT *, 'T CASE SF ...5.'
25 PRINT *, 'BBP MIN...2 VOLTS'
26 PRINT *, 'BBP SF...10VOLTS'
27 PRINT *, 'DO YOU WANT TO CHANGE THE DEFAULT MIN AND SF (Y/N)'
28 READ (*, '(A1)')ANS
29 IF(ANS.EQ. 'Y')THEN
30 PRINT *, 'INPUT THE NEW MIN AND SF PRT VALUES'
31 READ *, PRTMIN,PRTSF
32 PRINT *, 'INPUT THE NEW MIN AND SF ISOL VALUES'
33 READ *, TSOLMIN,TSOLSF
34 PRINT *, 'INPUT THE NEW MIN AND SF T CASE VALUES'
35 READ *, TCASMIN,TCASSF
36 PRINT *, 'INPUT THE NEW BBP MIN ANS SF VALUES'
37 READ *, BBRMIN,BBPSF
38 ELSEIF(ANS.EQ. 'N')THEN
39 GO TO 100
40 ELSEIF(ANS.NE. 'Y' .OR. ANS.NE. 'N')THEN
  
```

ORIGINAL PAGE IS  
OF POOR QUALITY.

87/04/30. 09.48.

74/860 DPT=1,ROUND= A/ S/ M/-D,-DS

FTN 5.1+642

```
1 41
1 42
1 43
1 44
1 45
1 46
1 47
1 48
1 49
1 50
1 51
1 52
1 53
1 54
1 55
1 56
1 57
1 58
1 59
1 60
1 61
1 62
1 63
1 64
1 65
1 66
1 67
1 68
1 69
1 70
1 71
1 72
1 73
1 74
1 75
1 76
1 77
1 78
1 79
1 80
1 81
1 82

SUBROUTINE SUB84
  PRINT *, 'WRONG INPUT'
  GO TO 5

  ENDIF
  Y1(NUM+1)=PRTMIN
  Y1(NUM+2)=PRTSF
  Y11(NUM+1)=88PMIN
  Y11(NUM+2)=88PSF
  Y5(NUM+1)=TSOLMIN
  Y5(NUM+2)=TSOLSF
  Y7(NUM+1)=TCASMIN
  Y7(NUM+2)=TCASSF
  CALL LEROY
  CALL CALPLT(2.,1.,-3)
  X(NUM+1)=0.
  X(NUM+2)=XSCALE
  CALL AXES(0.,0.,0.,8.,X(NUM+1),X(NUM+2),1.,10.,ISTRIN,.14,-18)
  CALL AXES(0.,0.,90.,5.,Y1(NUM+1),Y1(NUM+2),1.,10.,PRT,.14,3)
  CALL AXES(0.,5.,0.,8.,X(NUM+1),X(NUM+2),1.,10.,.1,0.0,1)
  CALL AXES(8.,0.,90.,5.,Y1(NUM+1),Y1(NUM+2),1.,10.,.1,0.0,-1)
  CALL NEWPEN(1)
  CALL LINPLT(X,Y1,NUM,1,0,0,1,1)
  CALL NEWPEN(2)
  CALL CALPLT(-1.,0.,-3)
  CALL AXES(0.,0.,90.,5.,Y11(NUM+1),Y11(NUM+2),1.,10.,'BBP',
    *.14,3)
  CALL CALPLT(1.,0.,-3)
  CALL LINPLT(X,Y11,NUM,1,0,0,1,2)
  CALL NEWPEN(3)
  CALL CALPLT(9.,0.,-3)
  CALL AXES(0.,0.,90.,5.,Y5(NUM+1),Y5(NUM+2),1.,10.,'ISOLATOR',
    *.14,8)
  CALL CALPLT(-9.,0.,-3)
  CALL LINPLT(X,Y5,NUM,1,0,0,1,3)
  CALL NEWPEN(4)
  CALL CALPLT(10.,0.,-3)
  CALL AXES(0.,0.,90.,5.,Y7(NUM+1),Y7(NUM+2),1.,10.,'CASE',
    *.14,4)
  CALL CALPLT(-10.,0.,-3)
  CALL LINPLT(X,Y7,NUM,1,0,0,1,3)
  CALL NEWPEN(1)
  CALL HEADR
  CALL NFRAME
```

83  
84RETURN  
END

---VARIABLE MAP---(LO=A)

---NAME---	---ADDRESS---	---BLOCK---	---PROPERTIES---	---SIZE---	---TYPE---	---NAME---	---ADDRESS---	---BLOCK---	---PROPERTIES---
ANS	0B	/TOP/			CHAR#1	PRTSF	1B	/KEY/	
BBPMIN	2B	/KEY/			REAL	RADMIN	13B	/KEY1/	
BBPSF	3B	/KEY/			REAL	RADSF	14B	/KEY1/	
BBRMIN	4B	/KEY/			REAL	TCASMIN	7B	/KEY1/	
BBRSF	5B	/KEY/			REAL	TCASSF	10B	/KEY1/	
EDAY	3B	/DATE/			REAL	TCHMIN	2B	/KEY1/	
EMNTH	2B	/DATE/			REAL	TCHSF	3B	/KEY1/	
FDAY	1B	/DATE/			REAL	TCKMIN	0B	/KEY1/	
FHR	1B	/TIM/			REAL	TCKSF	1B	/KEY1/	
FHRE	3B	/TIM/			REAL	TSOLMIN	4B	/KEY1/	
FMIN	2B	/TIM/			REAL	TSOLSF	5B	/KEY1/	
FMINF	4B	/TIM/			REAL	VPSMIN	11B	/KEY1/	
FMNTH	0B	/DATE/			REAL	VPSSF	12B	/KEY1/	
FYEAR	0B	/TIM/			REAL	X	5	DUMMY-ARG	
ISTRIN	0B	/TOP/			CHAR#18	XSCALE	0B	/SCALE/	
NUM	6B	/KEY1/			INTEGER	Y1	1	DUMMY-ARG	
PREMIN	15B	/KEY1/			REAL	Y11	4	DUMMY-ARG	
PRESF	16B	/KEY1/			REAL	Y5	2	DUMMY-ARG	
PRTMIN	0B	/KEY/			REAL	Y7	3	DUMMY-ARG	

ORIGINAL PAGE IS  
OF POOR QUALITY

---PROCEDURES---(LO=A)

---NAME---	---TYPE---	---ARGS---	---CLASS---	---NAME---	---TYPE---	---ARGS---	---CLASS---
AXES		11	SUBROUTINE	LINPLT		8	SUBROUTINE
CALPLT		3	SUBROUTINE	NEWPEN		1	SUBROUTINE
HEADR		0	SUBROUTINE	NFRAME		0	SUBROUTINE
LEROI		0	SUBROUTINE				

```

1  C
2  SUBROUTINE SUB5(Y1,Y10,Y8,X)
3
4  C
5  C*****THE FOLLOWING SUBROUTINE SUB5 PLOTS PRT,VPS AND**
6  C*****RADIOMETRIC DATA**
7  C
8  COMMON/TOP/ANS,ISTRIN
9  COMMON/SCALE/XSCALE
10 COMMON/TIM/YEAR,FHR,FMIN,FHRE,FHRE,FHRE
11 COMMON/DATE/FMINTH,FDAY,EMNTH,EDAY
12 COMMON/KEY/ PRTMIN,PRTSF,BBPMIN,BBPSF,BBPMIN,BBPSF
13 COMMON/KEY1/ TCKMIN,TCKSF,TCHMIN,TCHSF,TSOLMIN,TSOLSF,NUM,
14 *TCASMIN,TCASSF,VPSMIN,VPSSF,RADMIN,RADSF,PREMIN,PRESF
15 DIMENSION Y1(NUM+2),Y10(NUM+2),Y8(NUM+2),X(NUM+2)
16 CHARACTER ANS*1,ISTRIN*18
17 DATA ISTRIN/'ELAPSED TIME(HRS)'/
18 PRINT *, 'THE DEFAULT GRAPHING LIMITS ARE AS FOLLOWS : '
19 PRINT *, 'PRT MIN .....980 K'
20 PRINT *, 'PRT SF .....10'
21 PRINT *, 'VPS MIN .....0 VOLTS'
22 PRINT *, 'VPS SF .....(2.)'
23 PRINT *, 'RAD MIN .....(-2) C'
24 PRINT *, 'RAD SF .....2.'
25 PRINT *, 'DO YOU WANT TO CHANGE THE DEFAULT MIN AND SF?(Y/N)'
26
27 READ (*,'(A1)')ANS
28 IF(ANS.EQ. 'Y')THEN
29   PRINT *, 'INPUT THE NEW MIN AND SF PRT VALUES '
30   READ *,PRTMIN,PRTSF
31   PRINT *, 'INPUT THE NEW MIN AND SF VPS VALUES '
32   READ *,VPSMIN,VPSSF
33   PRINT *, 'INPUT THE NEW MIN AND SF RADIOMETRIC VAL'
34   READ *,RADMIN,RADSF
35   ELSEIF(ANS.EQ. 'N')THEN
36     GO TO 100
37   ELSEIF(ANS.NE. 'Y' .OR. ANS .NE. 'N')THEN
38     PRINT *, 'WRONG INPUT'
39     GO TO 5
40
41   ENDIF
42   Y1(NUM+1)=PRTMIN
43
44   100

```

```

41 Y1(NUM+2)=PRTSF
42 Y10(NUM+1)=VPSMIN
43 Y10(NUM+2)=VPSSF
44 Y8(NUM+1)=RADMIN
45 Y8(NUM+2)=RADSF
46 CALL LERDY
47 CALL CALPLT(2,1,-3)
48 X(NUM+1)=0.
49 X(NUM+2)=XSCALE
50 CALL AXES(0,0,0,9,X(NUM+1),X(NUM+2),1,10,ISTRIN,.14,-18)
51 CALL AXES(0,0,90,5,Y1(NUM+1),Y1(NUM+2),1,10,'PRT',.14,3)
52 CALL AXES(0,5,0,9,X(NUM+1),X(NUM+2),1,10,'',0,0,1)
53 CALL AXES(9,0,90,5,Y1(NUM+1),Y1(NUM+2),1,10,'',0,0,-1)
54 CALL NEWPEN (1)
55 CALL LINPLT(X,Y1,NUM,1,0,0,1,1)
56 CALL NEWPEN (2)
57 CALL CALPLT(-1,0,-3)
58 CALL AXES(0,0,90,5,Y10(NUM+1),Y10(NUM+2),1,10,'VPS',
59      *.14,3)
60 CALL CALPLT(1,0,0,-3)
61 CALL LINPLT(X,Y10,NUM,1,0,0,1,2)
62 CALL NEWPEN (3)
63 CALL CALPLT(10,0,-3)
64 CALL AXES(0,0,90,5,Y8(NUM+1),Y8(NUM+2),1,10,'RADIOMETRIC',
65      *.14,11)
66 CALL CALPLT(-10,0,-3)
67 CALL LINPLT(X,Y8,NUM,1,0,0,1,3)
68 CALL NEWPEN(1)
69 CALL HEADR
70 CALL NFRAME
71 RETURN
72 END

```

---VARIABLE MAP---(LO=A)

-NAME-----ADDRESS --BLOCK-----PROPERTIES-----TYPE-----SIZE

-NAME-----ADDRESS --BLOCK-----PROPERTIES

ANS	OB	/TOP/	EDAY	3B	/DATE/
BBPMIN	2B	/KEY/	EMNTH	2B	/DATE/
BBPSF	3B	/KEY/	FDAY	1B	/DATE/
BBRMIN	4B	/KEY/	FHR	1B	/TIM/
BBRSF	5B	/KEY/	FHRE	3B	/TIM/



SUBROUTINE SUB6 74/860 OPT=1,ROUND= A/ S/ M/-D,-DS FTN 5.1+642 87/04/30. 09.48.  
 DO=-LONG/-DT,ARG= COMMON/-FIXED,CS= USER/-FIXED,DB=-TB/-SB/-SL/-ER/-ID/-PMD/-ST,-AL,PL=5000  
 FTN5,I=HPLLOT,L=LF.

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
C
SUBROUTINE SUB6(Y11,Y12,Y4,Y6,X)
C
C*****THE FOLLOWING SUBROUTINE SUB6 PLOTS CIRCUIT TEMP
C*****CHAMBER TEMP BBP AND BBR*****
C
COMMON/TOP/ANS,ISTRIN
COMMON/SCALE/XSCALE
COMMON/TIM/FYEAR,FHR,FMIN,FHRE,FMINE
COMMON/DATE/FMINTH,FDAY,EMNTH,EDAY
COMMON/KEY/ PRTMIN,PRTSF,BBPMIN,BBPSF,BBRMIN,BBRSF
COMMON/KEY1/ TCKMIN,TCKSF,TCHMIN,TCHSF,TSOLMIN,TSOLSF,NUM,
*TCASMIN,TCASSF,VPSMIN,VPSSF,RADMIN,RADSF,PREMIN,PRESF
DIMENSION Y11(NUM+2),Y12(NUM+2),Y4(NUM+2),Y6(NUM+2),X(NUM+2)
CHARACTER ANS*1,ISTRIN*18
DATA ISTRIN/'ELAPSED TIME(HRS)'/
PRINT *, 'THE DEFAULT GRAPHING LIMITS ARE AS FOLLOWS : '
PRINT *, 'TCKMIN .....20 C'
PRINT *, 'TCKSF .....(5.)'
PRINT *, 'TCHMIN .....15 C'
PRINT *, 'TCHSF .....(5.)'
PRINT *, 'BBPMIN ..... 2 WATTS'
PRINT *, 'BBPSF .....2'
PRINT *, 'BBRMIN .....3.5 OHMS'
PRINT *, 'BBRSF .....(.5)'
PRINT *, 'DO YOU WANT TO CHANGE THE DEFAULT MIN AND SF(Y/N)?'
READ (*,'(A1)')ANS
IF(ANS.EQ.'Y')THEN
  PRINT *, 'INPUT THE NEW MIN AND SF TCK VALUES '
  READ *,TCKMIN,TCKSF
  PRINT *, 'INPUT THE NEW MIN AND SF TCH VALUES '
  READ *,TCHMIN,TCHSF
  PRINT *, 'INPUT THE NEW MIN AND SF BBP VALUES'
  READ *,BBPMIN,BBPSF
  PRINT *, 'INPUT THE NEW MIN AND SF BBR VALUES '
  READ *,BBRMIN,BBRSF
  ELSEIF(ANS.EQ.'N')THEN
    GO TO 100
  ELSEIF(ANS.NE.'Y'.OR.ANS.NE.'N')THEN
    PRINT *, 'WRONG INPUT'
  
```

```
1 41
1 42
43      GO TO 5
44      100
45      ENDIF
46      Y4(NUM+1)=TCKMIN
47      Y4(NUM+2)=TCKSF
48      Y11(NUM+1)=8BPMIN
49      Y11(NUM+2)=8BPSF
50      Y12(NUM+1)=8BRMIN
51      Y12(NUM+2)=8BRSF
52      Y6(NUM+1)=TCHMIN
53      Y6(NUM+2)=TCHSF
54      CALL LEROY
55      CALL CALPLT(2,1,-3)
56      X(NUM+1)=0.
57      X(NUM+2)=XSCALE
58      CALL AXES(0,0,0,9,X(NUM+1),X(NUM+2),1,10,ISTRIN,14,-18)
59      CALL AXES(0,0,90,5,Y4(NUM+1),Y4(NUM+2),1,10,
60      *CIRCUIT',14,7)
61      CALL AXES(0,5,0,9,X(NUM+1),X(NUM+2),1,10,'',0,0,1)
62      CALL AXES(9,0,90,5,Y4(NUM+1),Y4(NUM+2),1,10,'',0,0,-1)
63      CALL NEWPEN(1)
64      CALL LINPLT(X,Y4,NUM,1,0,0,1,1)
65      CALL NEWPEN(2)
66      CALL CALPLT(-1,0,-3)
67      CALL AXES(0,0,90,5,Y11(NUM+1),Y11(NUM+2),1,10,'BBP',
68      *.14,3)
69      CALL CALPLT(1,0,-3)
70      CALL LINPLT(X,Y11,NUM,1,0,0,1,2)
71      CALL NEWPEN(3)
72      CALL CALPLT(10,0,-3)
73      CALL AXES(0,0,90,5,Y12(NUM+1),Y12(NUM+2),1,10,'8BR',
74      *.14,3)
75      CALL CALPLT(-10,0,-3)
76      CALL LINPLT(X,Y12,NUM,1,0,0,1,3)
77      CALL NEWPEN(1)
78      CALL HEADR
79      CALL NFRAME
80      RETURN
81      END
```

11 11 11 11 11 11 11 11 11

```

41 X(NUM+1)=0.
42 X(NUM+2)=XSCALE
43 CALL AXES(0.,0.,0.,9.,X(NUM+1),X(NUM+2),1.,10.,ISTRIN,.14,-18)
44 CALL AXES(0.,0.,90.,5.,Y1(NUM+1),Y1(NUM+2),1.,10.,PRT,.14,3)
45 CALL AXES(0.,5.,0.,9.,X(NUM+1),X(NUM+2),1.,10.,.1.,0.0,1)
46 CALL AXES(9.,0.,90.,5.,Y1(NUM+1),Y1(NUM+2),1.,10.,.1.,0.0,-1)
47 CALL NEWPEN (1)
48 CALL LINPLT(X,Y1,NUM,1,0,0,1,1)
49 CALL NEWPEN (2)
50 CALL CALPLT(-1.,0.,-3)
51 CALL AXES(0.,0.,90.,5.,Y16(NUM+1),Y16(NUM+2),1.,10.,PRESSURE',
52 *.14,8)
53 CALL CALPLT(1.,0.,-3)
54 CALL LINPLT(X,Y16,NUM,1,0,0,1,2)
55 CALL NEWPEN(1)
56 CALL HEADR
57 CALL NFRAME
58 RETURN
59 END

```

---VARIABLE MAP---(LD=A)

---NAME---	ADDRESS	---BLOCK---	PROPERTIES	---TYPE---	SIZE	---NAME---	ADDRESS	---BLOCK---	PROPERTIES
ANS	OB	/TOP/		CHAR*1		PRTMIN	OB	/KEY/	
BBPMIN	2B	/KEY/	REAL			PRTSF	1B	/KEY/	
BBPSF	3B	/KEY/	REAL			RADMIN	13B	/KEY1/	
BBRMIN	4B	/KEY/	REAL			RADSF	14B	/KEY1/	
BBRSF	5B	/KEY/	REAL			TCASMIN	7B	/KEY1/	
EDAY	3B	/DATE/	REAL			TCASSF	10B	/KEY1/	
EMNTH	2B	/DATE/	REAL			TCHMIN	2B	/KEY1/	
FDAY	1B	/DATE/	REAL			TCHSF	3B	/KEY1/	
FHR	1B	/TIM/	REAL			TCKMIN	OB	/KEY1/	
FHRE	3B	/TIM/	REAL			TCKSF	1B	/KEY1/	
FMIN	2B	/TIM/	REAL			TSOLMIN	4B	/KEY1/	
FMINE	4B	/TIM/	REAL			TSOLSF	5B	/KEY1/	
FMNTH	OB	/DATE/	REAL			VPSMIN	11B	/KEY1/	
FYEAR	OB	/TIM/	REAL			VPSSF	12B	/KEY1/	
ISTRIN	OB	/TOP/	CHAR*18			X	3	DUMMY-ARG	
NUM	6B	/KEY1/	INTEGER			XSCALE	OB	/SCALE/	
PREMIN	15B	/KEY1/	REAL			Y1	1	DUMMY-ARG	
PRESF	16B	/KEY1/	REAL			Y16	2	DUMMY-ARG	



```
1 41 READ *,BBRMIN,BBRSF
1 42 ELSEIF(ANS.EQ.'N')THEN
1 43 GO TO 100
1 44 ELSEIF(ANS.NE.'Y'.OR. ANS .NE. 'N')THEN
1 45 PRINT *,'WRONG INPUT'
1 46 GO TO 5
1 47
1 48
100 48 DPRT(M+1)=PRTMIN
49 DPRT(M+2)=PRTSF
50 DBBP(M+1)=BBPMIN
51 DBBP(M+2)=BBPSF
52 DBBR(M+1)=BBRMIN
53 DBBR(M+2)=BBRSF
54 CALL LEROY
55 CALL CALPLT(2.,1.,-3)
56 DDAY(M+1)=DAYS-1.
57 DDAY(M+2)=6.
58 CALL AXES(0.,0.,0.,9.,DDAY(M+1),DDAY(M+2),1.,6.,ISTRIN,
59 *.14,-18)
60 WRITE(6,16)ISTRIN
61 FORMAT(1X,A18)
62 CALL AXES(0.,0.,90.,5.,DPRT(M+1),DPRT(M+2),1.,10.,'PRT',
63 *.14,3)
64 CALL AXES(0.,5.,0.,9.,DDAY(M+1),DDAY(M+2),1.,6.,' ',0.,0.,1)
65 CALL AXES(9.,0.,90.,5.,DPRT(M+1),DPRT(M+2),1.,10.,' ',0.,0.,-1)
66 CALL NEWPEN (1)
67 CALL LINPLT(DDAY,DPRT,M,1,0,0,1,1)
68 CALL NEWPEN (2)
69 CALL CALPLT(-1.,0.,-3)
70 CALL AXES(0.,0.,90.,5.,DBBP(M+1),DBBP(M+2),1.,10.,'BBP',
71 *.14,3)
72 CALL CALPLT(1.,0.,-3)
73 CALL LINPLT(DDAY,DBBP,M,1,0,0,1,2)
74 CALL NEWPEN (3)
75 CALL CALPLT(10.,0.,-3)
76 CALL AXES(0.,0.,90.,5.,DBBR(M+1),DBBR(M+2),1.,10.,'BBR',
77 *.14,3)
78 CALL CALPLT(-10.,0.,-3)
79 CALL LINPLT(DDAY,DBBR,M,1,0,0,1,3)
80 CALL NEWPEN (1)
81 CALL HEADR
82 CALL CHARACTER(2.5,7.75,.10,NSTRIN,0,14,.2)
```

ORIGINAL PAGE IS  
OF POOR QUALITY.

```

83 CALL NFRAME
84 PRINT *, 'THE DEFAULT GRAPHING LIMITS ARE AS FOLLOWS : '
85 PRINT *, 'PRT MIN ....980K'
86 PRINT *, 'PRT SF ....10 '
87 PRINT *, 'TCK MIN ....20'
88 PRINT *, 'TCK SF ....5.'
89 PRINT *, 'TCH MIN ....15'
90 PRINT *, 'TCH SF ....(5.)'
91 PRINT *, 'DO YOU WANT TO CHANGE THE DEFAULT MIN AND SF?(Y/N)'
92 READ (*, '(A1)')ANS
93 IF (ANS.EQ.'Y') THEN
94   PRINT *, 'INPUT THE NEW PRT MIN AND SF'
95   READ *, PRTMIN, PRTSF
96   PRINT *, 'INPUT THE NEW TCK MIN AND SF'
97   READ *, TCKMIN, TCKSF
98   PRINT *, 'INPUT THE NEW TCH MIN AND SF'
99   READ *, TCHMIN, TCHSF
100  ELSEIF (ANS.EQ.'N') THEN
101    GO TO 200
102  ELSEIF (ANS.NE.'Y' .OR. ANS .NE. 'N') THEN
103    PRINT *, 'WRONG INPUT'
104    GO TO 15
105  ENDIF
106  DCKT(M+1)=TCKMIN
107  DCKT(M+2)=TCKSF
108  DCHBR(M+1)=TCHMIN
109  DCHBR(M+2)=TCHSF
110  DPRT(M+1)=PRTMIN
111  DPRT(M+2)=PRTSF
112  DDAY(M+1)=DAYS-1.
113  DDAY(M+2)=6.
114  CALL LEROY
115  CALL CALPLT(2., 1., -3)
116  DDAY(M+1)=DAYS-1.
117  DDAY(M+2)=6.
118  WRITE(6, 16) ISTRIN
119  CALL AXES(0., 0., 0., 9., DDAY(M+1), DDAY(M+2), 1., 6.,
120  * 'ELAPSED TIMES(DAYS)', .14, -20)
121  WRITE(6, 16) ISTRIN
122  CALL AXES(0., 0., 90., 5., DPRT(M+1), DPRT(M+2), 1., 10., 'PRT',
123  * .14, 3)
124  CALL AXES(0., 5., 0., 9., DDAY(M+1), DDAY(M+2), 1., 6., ' ', 0., 0., 1)

```

```

125 CALL AXES(9.0,0.0,90.0,5.0,DPRT(M+1),DPRT(M+2),1.0,10.0,'',0.0,0,-1)
126 CALL NEWPEN (1)
127 CALL LINPLT(DDAY,DPRT,M,1,0,0,1,1)
128 CALL NEWPEN (2)
129 CALL CALPLT(-1.0,0.0,-3)
130 CALL AXES(0.0,0.0,90.0,5.0,DCKT(M+1),DCKT(M+2),1.0,10.0,'CIRCUIT',
131 *14,7)
132 CALL CALPLT(1.0,0.0,-3)
133 CALL LINPLT(DDAY,DCKT,M,1,0,0,1,2)
134 CALL NEWPEN (3)
135 CALL CALPLT(10.0,0.0,-3)
136 CALL AXES(0.0,0.0,90.0,5.0,DCHBR(M+1),DCHBR(M+2),1.0,10.0,'CHBR',
137 *14,4)
138 CALL CALPLT(-10.0,0.0,-3)
139 CALL LINPLT(DDAY,DCHBR,M,1,0,0,1,3)
140 CALL NEWPEN (1)
141 CALL HEADR
142 CALL CHARACTER(2.5,7.75,10,NSTRIN,0,14,0,2)
143 CALL NFRAME
144 PRINT *, 'THE DEFAULT GRAPHING LIMITS ARE AS FOLLOWS : '
145 PRINT *, 'PRT MIN ...980K'
146 PRINT *, 'PRT SF ...10'
147 PRINT *, 'ISOL MIN ...110'
148 PRINT *, 'ISOL SF ...5'
149 PRINT *, 'CASE MIN ...115'
150 PRINT *, 'CASE SF ...5'
151 PRINT *, 'DO YOU WANT TO CHANGE THE DEFAULT MIN AND SF?(Y/N)'
152 READ (*, '(A1)')ANS
153 IF(ANS.EQ. 'Y')THEN
154     PRINT *, 'INPUT THE NEW PRT MIN AND SF'
155     READ *, PRTMIN,PRTSF
156     PRINT *, 'INPUT THE NEW ISOL MIN AND SF'
157     READ *, TSOLMIN,TSOLSF
158     PRINT *, 'INPUT THE NEW CASE MIN AND SF'
159     READ *, TCASMIN,TCASSF
160     ELSEIF(ANS.EQ. 'N')THEN
161         GO TO 300
162     ELSEIF(ANS.NE. 'Y' .OR. ANS .NE. 'N')THEN
163         PRINT *, 'WRONG INPUT'
164         GO TO 25
165     ENDIF
166 DPRT(M+1)=PRTMIN

```

25

300



```

167 DPRT(M+2)=PRTSF
168 DISOL(M+1)=TSOLMIN
169 DISOL(M+2)=TSOLSF
170 DCASE(M+1)=TCASHMIN
171 DCASE(M+2)=TCASSF
172 CALL LEROY
173 CALL CALPLT(2,1,-3)
174 DDAY(M+1)=DAYS-1.
175 DDAY(M+2)=6.
176 CALL AXES(0,0,0,9,DDAY(M+1),DDAY(M+2),1,6,
177 *ELAPSED TIMES(DAYS$)',14,-20)
178 CALL AXES(0,0,90,5,DPRT(M+1),DPRT(M+2),1,10,'PRT',
179 *.14,3)
180 CALL AXES(0,5,0,9,DDAY(M+1),DDAY(M+2),1,6,'',0,0,1)
181 CALL AXES(9,0,90,5,DPRT(M+1),DPRT(M+2),1,10,'',0,0,-1)
182 CALL NEWPEN (1)
183 CALL LINPLT(DDAY,DPRT,M,1,0,0,1,1)
184 CALL NEWPEN (2)
185 CALL CALPLT(-1,0,-3)
186 CALL AXES(0,0,90,5,DISOL(M+1),DISOL(M+2),1,10,'ISOLATOR',
187 *.14,8)
188 CALL CALPLT(1,0,-3)
189 CALL LINPLT(DDAY,DISOL,M,1,0,0,1,2)
190 CALL NEWPEN (3)
191 CALL CALPLT(10,0,-3)
192 CALL AXES(0,0,90,5,DCASE(M+1),DCASE(M+2),1,10,'CASE',
193 *.14,4)
194 CALL CALPLT(-10,0,-3)
195 CALL LINPLT(DDAY,DCASE,M,1,0,0,1,3)
196 CALL NEWPEN(1)
197 CALL HEADR
198 CALL CHARACTER(2.5,7.75,10,NSTRIN,0,14,.2)
199 CALL NFRAME
200 PRINT *,THE DEFAULT GRAPHING LIMITS ARE AS FOLLOWS :
201 PRINT *,PRT MIN ....980K
202 PRINT *,PRT SF ....10
203 PRINT *,RAD MIN ...0
204 PRINT *,RAD SF ...2
205 PRINT *,VPS MIN ...0
206 PRINT *,VPS SF ...2
207 PRINT *,DO YOU WANT TO CHANGE THE DEFAULT MIN AND SF?(Y/N)
208 READ (*, '(A1)')ANS

```

```

209 IF(ANS.EQ.'Y')THEN
210   PRINT *, 'INPUT THE NEW PRT MIN ANS SF'
211   READ *,PRTMIN,PRTSF
212   PRINT *, 'INPUT THE NEW RAD MIN AND SF'
213   READ *,RADMIN,RADSF
214   PRINT *, 'INPUT THE NEW VPS MIN AND SF'
215   READ *,VPSMIN,VPSSF
216   ELSEIF(ANS.EQ.'N')THEN
217     GO TO 400
218   ELSEIF(ANS.NE.'Y'.OR. ANS .NE. 'N')THEN
219     PRINT *, 'WRONG INPUT'
220     GO TO 35
221   ENDIF
222   DPRT(M+1)=PRTMIN
223   DPRT(M+2)=PRTSF
224   DRAD(M+1)=RADMIN
225   DRAD(M+2)=RADSF
226   DVPS(M+1)=VPSMIN
227   DVPS(M+2)=VPSSF
228   CALL LEROY
229   CALL CALPLT(2.,1.,-3)
230   DDAY(M+1)=DAYS-1.
231   DDAY(M+2)=6.
232   CALL AXES(0.,0.,0.,9.,DDAY(M+1),DDAY(M+2),1.,6.,
233     * 'ELAPSED TIMES(DAYS$)',.14,-20)
234   CALL AXES(0.,0.,90.,5.,DPRT(M+1),DPRT(M+2),1.,10.,'PRT',
235     *.14,3)
236   CALL AXES(0.,5.,0.,9.,DDAY(M+1),DDAY(M+2),1.,6., '0.0,1)
237   CALL AXES(9.,0.,90.,5.,DPRT(M+1),DPRT(M+2),1.,10., '0.0,-1)
238   CALL NEWPEN (1)
239   CALL LINPLT(DDAY,DPRT,M,1,0,0,1,1)
240   CALL NEWPEN (2)
241   CALL CALPLT(-1.,0.,-3)
242   CALL AXES(0.,0.,90.,5.,DRAD(M+1),DRAD(M+2),1.,10., 'RADIOMETRIC',
243     *.14,11)
244   CALL CALPLT(1.,0.,-3)
245   CALL LINPLT(DDAY,DRAD,M,1,0,0,1,2)
246   CALL NEWPEN (3)
247   CALL CALPLT(10.,0.,-3)
248   CALL AXES(0.,0.,90.,5.,DVPS(M+1),DVPS(M+2),1.,10., 'VPS',
249     *.14,3)
250   CALL CALPLT(-10.,0.,-3)

```

87/04/30. 09.48.

FTN 5.1+642

74/860 OPT=1,ROUND= A/ S/ M/-D,-DS

SUBROUTINE SUB8

ORIGINAL PAGE IS  
OF POOR QUALITY

```

251 CALL LINPLT(DDAY,DVPS,M,1,0,0,1,3)
252 CALL NEWPEN(1)
253 CALL HEADR
254 CALL CHARACTER(2,5,7,75,10,NSTRIN,0,14,0,2)
255 CALL NFRAME
256 PRINT *, 'THE DEFAULT GRAPHING LIMITS ARE AS FOLLOWS : '
257 PRINT *, 'PRT MIN ....980K'
258 PRINT *, 'PRT SF ....10 '
259 PRINT *, 'PRE MIN ....01'
260 PRINT *, 'PRE SF ....4.'
261 PRINT *, 'DO YOU WANT TO CHANGE THE DEFAULT MIN AND SF?(Y/N)'
262 READ (*, '(A1)')ANS
263 IF(ANS.EQ.'Y')THEN
264   PRINT *, 'INPUT THE NEW PRT MIN AND SF'
265   READ *, PRTMIN,PRTSF
266   PRINT *, 'INPUT THE NEW PRE MIN AND SF'
267   READ *, PREMIN,PRESF
268   ELSEIF(ANS.EQ.'N')THEN
269     GO TO 500
270   ELSEIF(ANS.NE.'Y'.OR. ANS.NE.'N')THEN
271     PRINT *, 'WRONG INPUT'
272     GO TO 45
273   ENDIF
274   DPRT(M+1)=PRTMIN
275   DPRT(M+2)=PRTSF
276   DPRESS(M+1)=PREMIN
277   DPRESS(M+2)=PRESF
278   CALL LEROY
279   CALL CALPLT(2,1,0,-3)
280   DDAY(M+1)=DAYS-1.
281   DDAY(M+2)=6.
282   CALL AXES(0,0,0,0,9,DDAY(M+1),DDAY(M+2),1,0,6,0)
283   *ELAPSED TIMES(DAYS),14,-20)
284   CALL AXES(0,0,0,90,5,DPRT(M+1),DPRT(M+2),1,0,10,0,'PRT',
285     *14,3)
286   CALL AXES(0,5,0,0,9,DDAY(M+1),DDAY(M+2),1,0,6,0,'0,0,1)
287   CALL AXES(9,0,0,90,5,DPRT(M+1),DPRT(M+2),1,0,10,0,'0,0,0,-1)
288   CALL NEWPEN (1)
289   CALL LINPLT(DDAY,DPRT,M,1,0,0,1,1)
290   CALL NEWPEN (2)
291   CALL CALPLT(-1,0,0,-3)
292   CALL AXES(0,0,0,90,5,DPRESS(M+1),DPRESS(M+2),1,0,10,0)

```

```
293 *'PRESSURE',.14,8)
294 CALL CALPLT(1.,0.,-3)
295 CALL LINPLT(DDAY,DPRESS,M,1,0,0,1,2)
296 CALL NEWPEN (1)
297 CALL HEADR
298 CALL CHARACTER(2.5,7.75,.10,NSTRIN,0,14,.2)
299 CALL NFRAME
300 RETURN
301 END
```

## ---VARIABLE MAP---(LO=A)

-NAME---ADDRESS --BLOCK-----PROPERTIES-----TYPE-----SIZE

ANS	OB	/TOP/	CHAR#1		
BBPMIN	2B	/KEY/	REAL		
BBPSF	3B	/KEY/	REAL		
BBRMIN	4B	/KEY/	REAL		
BBRSF	5B	/KEY/	REAL		
DAYE	2513B		REAL		*S*
DAYS	2512B		REAL		
DBBP	3	DUMMY-ARG	REAL	ADJ-ARY	
DBBR	4	DUMMY-ARG	REAL	ADJ-ARY	
DCASE	8	DUMMY-ARG	REAL	ADJ-ARY	
DCHBR	7	DUMMY-ARG	REAL	ADJ-ARY	
DCKT	5	DUMMY-ARG	REAL	ADJ-ARY	
DDAY	1	DUMMY-ARG	REAL	ADJ-ARY	
DISOL	6	DUMMY-ARG	REAL	ADJ-ARY	
DPRESS	11	DUMMY-ARG	REAL	ADJ-ARY	
DPRT	2	DUMMY-ARG	REAL	ADJ-ARY	
DRAD	9	DUMMY-ARG	REAL	ADJ-ARY	
DVPS	10	DUMMY-ARG	REAL	ADJ-ARY	
EDAY	3B	/DATE/	REAL		
EMNTH	2B	/DATE/	REAL		
FDAY	1B	/DATE/	REAL		
FHR	1B	/TIM/	REAL		
FHRE	3B	/TIM/	REAL		
FMIN	2B	/TIM/	REAL		
FMINE	4B	/TIM/	REAL		
FMNTH	OB				
FYEAR	OB				
IDAYE	14	DUMMY-ARG			
IDAYS	13	DUMMY-ARG			
ISTRIN	OB	/TOP/			
M	12	DUMMY-ARG			
NSTRIN	2510B				
NUM	6B	/KEY1/			
PREMIN	15B	/KEY1/			
PRESF	16B	/KEY1/			
PRTMIN	OB	/KEY/			
PRTSF	1B	/KEY/			
RADMIN	13B	/KEY1/			
RADSF	14B	/KEY1/			
TCASMIN	7B	/KEY1/			
TCASSF	10B	/KEY1/			
TCHMIN	2B	/KEY1/			
TCHSF	3B	/KEY1/			
TCKMIN	OB	/KEY1/			
TCKSF	1B	/KEY1/			
TSOLMIN	4B	/KEY1/			
TSOLSF	5B	/KEY1/			
VPSMIN	11B	/KEY1/			
VPSSF	12B	/KEY1/			
XSCALE	OB	/SCALE/			

ORIGINAL PAGE IS  
OF POOR QUALITY

SUBROUTINE SUB9  
DO=-LONG/-OT,ARG= COMMON/-FIXED,CS= USER/-FIXED,DB=-TB/-SB/-SL/-ER/-ID/-PMD/-ST,-AL,PL=5000  
FTN5,I=HPLOT,L=LF.

74/860 OPT=1,ROUND= A/ S/ M/-D,-DS FTN 5.1+642 87/04/30. 09.48.

```
1 C
2
3
4 C
5 SUBROUTINE SUB9(WWK,WPRT,WB8P,WB8R,WCKT,WISOL,
6
7 *THIS SUBROUTINE PLOTS THE WEEKLY AVERAGES OF ALL
8 *PARAMETERS
9
10 * WCHBR,WCASE,WRAD,WVPS,WPRESS,NW)
11 COMMON/TP/ANS,ISTRIN
12 COMMON/SCALE/XSCALE
13 COMMON/TIM/FYEAR,FHR,FMIN,FHRE,FMINE
14 COMMON/DATE/FMNTN,FDAY,EMNTH,EDAY
15 COMMON/KEY/ PRTMIN,PRTSF,BBPMIN,BBPSF,BBRMIN,BBRSF
16 COMMON/KEY1/ TCKMIN,TCKSF,TCHMIN,TCHSF,TSOLMIN,TSOLSF,NUM,
17 TCASMIN,TCASSF,VPSPMIN,VPSSF,RADMIN,RADSF,PREMIN,PRESF
18 DIMENSION WWK(NW+2),WPRT(NW+2),WBBP(NW+2),WBBR(NW+2)
19 DIMENSION WCKT(NW+2),WISOL(NW+2),WCHBR(NW+2),WCASE(NW+2),
20 WRAD(NW+2),WVPS(NW+2),WPRESS(NW+2)
21 CHARACTER ISTRIN*19,ANS*1,NSTRIN*14
22 DATA NSTRIN/'WEEKLY AVERAGE'/
23 ISTRIN='ELAPSED TIME(WEEKS)'
24 DAYS=FLOAT(IDAYS)
25 DAYE=FLOAT(IDAYE)
26 PRINT *, 'THE DEFAULT GRAPHING LIMITS ARE AS FOLLOWS : '
27 PRINT *, 'PRT MIN ....980K'
28 PRINT *, 'PRT SF ....10'
29 PRINT *, 'BBP MIN ...2 WATTS'
30 PRINT *, 'BBP SF ...2'
31 PRINT *, 'BBR MIN ...2.5 OHMS'
32 PRINT *, 'BBR SF ...(.5)'
33 PRINT *, 'DO YOU WANT TO CHANGE THE DEFAULT MIN AND SF?(Y/N)'
34 READ (*, '(A1)')ANS
35 IF(ANS.EQ.'Y')THEN
36 PRINT *, 'INPUT THE NEW PRT MIN ANS SF'
37 READ *,PRTMIN,PRTSF
38 PRINT *, 'INPUT THE NEW BBP MIN AND SF'
39 READ *,BBPMIN,BBPSF
40 PRINT *, 'INPUT THE NEW BBR MIN AND SF'
```

```

1  41 READ *,BBRMIN,BBRSF
1  42 ELSEIF(ANS.EQ.'N')THEN
1  43   GO TO 100
1  44   ELSEIF(ANS.NE.'Y'.OR.ANS.NE.'N')THEN
1  45     PRINT *,'WRONG INPUT'
1  46     GO TO 5
1  47   ENDIF
100 48   WPRT(NW+1)=PRTMIN
49   WPRT(NW+2)=PRTSF
50   WBBP(NW+1)=BBPMIN
51   WBBP(NW+2)=BBPSF
52   WBBR(NW+1)=BBRMIN
53   WBBR(NW+2)=BBRSF
54   CALL LEROY
55   CALL CALPLT(2,1,-3)
56   WWK(NW+1)=0.
57   WWK(NW+2)=1.
58   CALL AXES(0,0,0,9,WWK(NW+1),WWK(NW+2),1,6,ISTRIN,
59   *14,-19)
60   CALL AXES(0,0,90,5,WPRT(NW+1),WPRT(NW+2),1,10,'PRT',
61   *14,3)
62   CALL AXES(0,5,0,9,WWK(NW+1),WWK(NW+2),1,6,'',0,0,1)
63   CALL AXES(9,0,90,5,WPRT(NW+1),WPRT(NW+2),1,10,'',0,0,-1)
64   CALL NEWPEN (1)
65   CALL LINPLT(WWK,WPRT,NW,1,0,0,1,1)
66   CALL NEWPEN (2)
67   CALL CALPLT(-1,0,-3)
68   CALL AXES(0,0,90,5,WBBP(NW+1),WBBP(NW+2),1,10,'BBP',
69   *14,3)
70   CALL CALPLT(1,0,0,-3)
71   CALL LINPLT(WWK,WBBP,NW,1,0,0,1,2)
72   CALL NEWPEN (3)
73   CALL CALPLT(10,0,0,-3)
74   CALL AXES(0,0,90,5,WBBR(NW+1),WBBR(NW+2),1,10,'BBR',
75   *14,3)
76   CALL CALPLT(-10,0,0,-3)
77   CALL LINPLT(WWK,WBBR,NW,1,0,0,1,3)
78   CALL NEWPEN (1)
79   CALL HEADR
80   CALL CHARACTER(2,5,7,75,10,NSTRIN,0,14,0,2)
81   CALL NFRAME
82   PRINT *,'THE DEFAULT GRAPHING LIMITS ARE AS FOLLOWS : '

```

ORIGINAL PAGE IS  
OF POOR QUALITY

```

83 PRINT *, 'PRT MIN ....980K'
84 PRINT *, 'PRT SF ....10'
85 PRINT *, 'TCK MIN ...20'
86 PRINT *, 'TCK SF ...5'
87 PRINT *, 'TCH MIN ...15'
88 PRINT *, 'TCH SF ...5'
89 PRINT *, 'DO YOU WANT TO CHANGE THE DEFAULT MIN AND SF?(Y/N)'
90 READ (*, '(A1)')ANS
91 IF (ANS.EQ.'Y') THEN
92   PRINT *, 'INPUT THE NEW PRT MIN ANS SF'
93   READ *, PRTMIN, PRTSF
94   PRINT *, 'INPUT THE NEW TCK MIN AND SF'
95   READ *, TCKMIN, TCKSF
96   PRINT *, 'INPUT THE NEW TCH MIN AND SF'
97   READ *, TCHMIN, TCHSF
98   ELSEIF (ANS.EQ.'N') THEN
99     GO TO 200
100  ELSEIF (ANS.NE.'Y' .OR. ANS .NE. 'N') THEN
101    PRINT *, 'WRONG INPUT'
102    GO TO 15
103  ENDIF
104  WCKT(NW+1)=TCKMIN
105  WCKT(NW+2)=TCKSF
106  WCHBR(NW+1)=TCHMIN
107  WCHBR(NW+2)=TCHSF
108  WPRT(NW+1)=PRTMIN
109  WPRT(NW+2)=PRTSF
110  WWK(NW+1)=0.
111  WWK(NW+2)=1.
112  CALL LEROY
113  CALL CALPLT(2., 1., -3)
114  WWK(NW+2)=1.
115  CALL AXES(0., 0., 0., 9., WWK(NW+1), WWK(NW+2), 1., 6.,
116  *ELAPSED TIMES(WEEKS$), 14, -21)
117  CALL AXES(0., 0., 90., 5., WPRT(NW+1), WPRT(NW+2), 1., 10., 'PRT',
118  *14, 3)
119  CALL AXES(0., 5., 0., 9., WWK(NW+1), WWK(NW+2), 1., 6., '0.0, 1)
120  CALL AXES(9., 0., 90., 5., WPRT(NW+1), WPRT(NW+2), 1., 10., '0.0, -1)
121  CALL NEWPEN (1)
122  CALL LINPLT(WWK, WPRT, NW, 1, 0, 0, 1, 1)
123  CALL NEWPEN (2)
124  CALL CALPLT(-1., 0., -3)

```

ORIGINAL PAGE IS  
OF POOR QUALITY

```

125 CALL AXES(0,0,0,90,5,WCKT(NW+1),WCKT(NW+2),1,10,10,'CIRCUIT',
126 *14,7)
127 CALL CALPLT(1,0,0,-3)
128 CALL LINPLT(WWK,WCKT,NW,1,0,0,1,2)
129 CALL NEWPEN (3)
130 CALL CALPLT(10,0,0,-3)
131 CALL AXES(0,0,0,90,5,WCHBR(NW+1),WCHBR(NW+2),1,10,10,'CHAMBER',
132 *14,7)
133 CALL CALPLT(-10,0,0,-3)
134 CALL LINPLT(WWK,WCHBR,NW,1,0,0,1,3)
135 CALL NEWPEN (1)
136 CALL HEADR
137 CALL CHARACTER(2.5,7.75,10,NSTRIN,0,14,0,2)
138 CALL NFRAME
139 PRINT *, 'THE DEFAULT GRAPHING LIMITS ARE AS FOLLOWS : '
140 PRINT *, 'PRT MIN ...980K'
141 PRINT *, 'PRT SF ...10 '
142 PRINT *, 'ISOL MIN ...110'
143 PRINT *, 'ISOL SF ...5'
144 PRINT *, 'CASE MIN ...115'
145 PRINT *, 'CASE SF ...5'
146 PRINT *, 'DO YOU WANT TO CHANGE THE DEFAULT MIN AND SF?(Y/N)'
147 READ (*, '(A1)')ANS
148 IF(ANS.EQ.'Y')THEN
149   PRINT *, 'INPUT THE NEW PRT MIN AND SF'
150   READ *, PRTMIN,PRTSF
151   PRINT *, 'INPUT THE NEW ISOL MIN AND SF'
152   READ *, TSOLMIN,TSOLSF
153   PRINT *, 'INPUT THE NEW CASE MIN AND SF'
154   READ *, TCASMIN,TCASSF
155   ELSEIF(ANS.EQ.'N')THEN
156     GO TO 300
157   ELSEIF(ANS.NE.'Y' .OR. ANS .NE. 'N')THEN
158     PRINT *, 'WRONG INPUT'
159     GO TO 25
160   ENDIF
161   WPRT(NW+1)=PRTMIN
162   WPRT(NW+2)=PRTSF
163   WISOL(NW+1)=TSOLMIN
164   WISOL(NW+2)=TSOLSF
165   WCASE(NW+1)=TCASMIN
166   WCASE(NW+2)=TCASSF

```

25

300



ORIGINAL PAGE IS  
OF POOR QUALITY

```

167 CALL LEROY
168 CALL CALPLT(2.,1.,-3)
169 WWK(NW+1)=0.
170 WWK(NW+2)=1.
171 CALL AXES(0.,0.,0.,9.,WWK(NW+1),WWK(NW+2),1.,6.,
172 *'ELAPSED TIMES(DAYS)',.14,-21)
173 CALL AXES(0.,0.,90.,5.,WPRT(NW+1),WPRT(NW+2),1.,10.,'PRT',
174 *.14,3)
175 CALL AXES(0.,5.,0.,9.,WWK(NW+1),WWK(NW+2),1.,6.,' ',0.0,1)
176 CALL AXES(9.,0.,90.,5.,WPRT(NW+1),WPRT(NW+2),1.,10.,' ',0.0,-1)
177 CALL NEWPEN (1)
178 CALL LINPLT(WWK,WPRT,NW,1,0,0,1,1)
179 CALL NEWPEN (2)
180 CALL CALPLT(-1.,0.,-3)
181 CALL AXES(0.,0.,90.,5.,WISOL(NW+1),WISOL(NW+2),1.,10.,'ISOLATOR',
182 *.14,8)
183 CALL CALPLT(1.,0.,-3)
184 CALL LINPLT(WWK,WISOL,NW,1,0,0,1,2)
185 CALL NEWPEN (3)
186 CALL CALPLT(10.,0.,-3)
187 CALL AXES(0.,0.,90.,5.,WCASE(NW+1),WCASE(NW+2),1.,10.,'CASE',
188 *.14,4)
189 CALL CALPLT(-10.,0.,-3)
190 CALL LINPLT(WWK,WCASE,NW,1,0,0,1,3)
191 CALL NEWPEN(1)
192 CALL HEADR
193 CALL CHARACTER(2.5,7.75,.10,NSTRIN,0,14,.2)
194 CALL NFRAME
195 PRINT *, 'THE DEFAULT GRAPHING LIMITS ARE AS FOLLOWS : '
196 PRINT *, 'PRT MIN ...980K'
197 PRINT *, 'PRT SF ...10 '
198 PRINT *, 'RAD MIN ...0 '
199 PRINT *, 'RAD SF ...2.'
200 PRINT *, 'VPS MIN ...0.'
201 PRINT *, 'VPS SF ...(.2.)'
202 PRINT *, 'DO YOU WANT TO CHANGE THE DEFAULT MIN AND SF?(Y/N)'
203 READ (*, '(A1)')ANS
204 IF(ANS.EQ.'Y')THEN
205   PRINT *, 'INPUT THE NEW PRT MIN ANS SF'
206   READ *,PRTMIN,PRTSF
207   PRINT *, 'INPUT THE NEW RAD MIN AND SF'
208   READ *,RADMIN,RADSF

```

35

1  
1  
1  
1

SUBROUTINE SUB9

74/860 OPT=1,ROUND=A/ S/ M/-D,-DS

FTN 5.1+642

87/04/30. 09.48.

```

1 209 PRINT *, 'INPUT THE NEW VPS MIN AND SF'
1 210 READ *, VPSMIN, VPSSF
1 211 ELSEIF(ANS.EQ.'N') THEN
1 212 GO TO 400
1 213 ELSEIF(ANS.NE.'Y' .OR. ANS .NE. 'N') THEN
1 214 PRINT *, 'WRONG INPUT'
1 215 GO TO 35
1 216
1 217 ENDIF
218 WVRT(NW+1)=PRTMIN
219 WVRT(NW+2)=PRTSF
220 WRAD(NW+1)=RADMIN
221 WRAD(NW+2)=RADSF
222 WVPS(NW+1)=VPSMIN
223 WVPS(NW+2)=VPSSF
224 CALL LEROY
225 CALL CALPLT(2,1,-3)
226 WVK(NW+1)=0.
227 WVK(NW+2)=2.
228 CALL AXES(0,0,0,0,9,WVK(NW+1),WVK(NW+2),1,6,
229 *ELAPSED TIMES(WEEKS)),14,-21)
230 CALL AXES(0,0,0,90,5,WVRT(NW+1),WVRT(NW+2),1,10,'PRT',
231 *14,3)
232 CALL AXES(0,5,0,9,WVK(NW+1),WVK(NW+2),1,6,0,0,1)
233 CALL AXES(9,0,90,5,WVRT(NW+1),WVRT(NW+2),1,10,0,0,-1)
234 CALL NEWPEN (1)
235 CALL LINPLT(WVK,WVRT,NW,1,0,0,1,1)
236 CALL NEWPEN (2)
237 CALL CALPLT(-1,0,0,-3)
238 CALL AXES(0,0,0,90,5,WRAD(NW+1),WRAD(NW+2),1,10,'RADIOMETRIC',
239 *14,11)
240 CALL CALPLT(1,0,0,-3)
241 CALL LINPLT(WVK,WRAD,NW,1,0,0,1,2)
242 CALL NEWPEN (3)
243 CALL CALPLT(10,0,0,-3)
244 CALL AXES(0,0,0,90,5,WVPS(NW+1),WVPS(NW+2),1,10,'VPS',
245 *14,3)
246 CALL CALPLT(-10,0,0,-3)
247 CALL LINPLT(WVK,WVPS,NW,1,0,0,1,3)
248 CALL NEWPEN(1)
249 CALL HEADR
250 CALL CHARACTER(2.5,7.75,10,NSTRIN,0,14,2)
CALL NFRAME

```

ORIGINAL PAGE IS  
OF POOR QUALITY

```

251 PRINT *, 'THE DEFAULT GRAPHING LIMITS ARE AS FOLLOWS : '
252 PRINT *, 'PRT MIN ...980K'
253 PRINT *, 'PRT SF ...10 '
254 PRINT *, 'PRE MIN ...-.01'
255 PRINT *, 'PRE SF ...4.'
256 PRINT *, 'DO YOU WANT TO CHANGE THE DEFAULT MIN AND SF?(Y/N)'
257 READ (*, '(A1)')ANS
258 IF (ANS.EQ.'Y') THEN
259   PRINT *, 'INPUT THE NEW PRT MIN AND SF'
260   READ *, PRTMIN, PRTSF
261   PRINT *, 'INPUT THE NEW PRE MIN AND SF'
262   READ *, PREMIN, PRESF
263   ELSEIF (ANS.EQ.'N') THEN
264     GO TO 500
265   ELSEIF (ANS.NE.'Y'.OR. ANS .NE. 'N') THEN
266     PRINT *, 'WRONG INPUT'
267     GO TO 45
268
269   ENDIF
270   WPRT(NW+1)=PRTMIN
271   WPRT(NW+2)=PRTSF
272   WPRESS(NW+1)=PREMIN
273   WPRESS(NW+2)=PRESF
274   CALL LEROY
275   CALL CALPLT(2.,1.,-3)
276   WWK(NW+1)=0.
277   WWK(NW+2)=1.
278   CALL AXES(0.,0.,0.,9.,WWK(NW+1),WWK(NW+2),1.,6.,
279   *'ELAPSED TIMES(WEEKS$)',.14,-21)
280   CALL AXES(0.,0.,90.,5.,WPRT(NW+1),WPRT(NW+2),1.,10.,'PRT',
281   *.14,3)
282   CALL AXES(0.,5.,0.,9.,WWK(NW+1),WWK(NW+2),1.,6.,'0.0,1)
283   CALL AXES(9.,0.,90.,5.,WPRT(NW+1),WPRT(NW+2),1.,10.,'0.0,-1)
284   CALL NEWPEN (1)
285   CALL LINPLT(WWK,WPRT,NW,1,0,0,1,1)
286   CALL NEWPEN (2)
287   CALL CALPLT(-1.,0.,-3)
288   CALL AXES(0.,0.,90.,5.,WPRESS(NW+1),WPRESS(NW+2),1.,10.,
289   *'PRESSURE',.14,8)
290   CALL CALPLT(1.,0.,-3)
291   CALL LINPLT(WWK,WPRESS,NW,1,0,0,1,2)
292   CALL NEWPEN (1)
293   CALL HEADR

```

293  
294  
295  
296

CALL CHARACT(2.5,7.75,.10,NSTRIN,0,14,.2)  
CALL NFRAME  
RETURN  
END

---VARIABLE MAP---(LO=A)

-NAME-----ADDRESS --BLOCK-----PROPERTIES-----TYPE-----SIZE

ANS	OB	/TOP/	CHAR#1			PRTSF	1B	/KEY/
BBPMIN	2B	/KEY/	REAL			RADMIN	13B	/KEY1/
BBPSF	3B	/KEY/	REAL			RADSF	14B	/KEY1/
BBRMIN	4B	/KEY/	REAL			TCASMIN	7B	/KEY1/
BBRSF	5B	/KEY/	REAL			TCASSF	10B	/KEY1/
DAYE	2510B		REAL	*S*		TCHMIN	2B	/KEY1/
DAYS	2506B		REAL	*S*		TCHSF	3B	/KEY1/
EDAY	3B	/DATE/	REAL			TCKMIN	0B	/KEY1/
EMNTH	2B	/DATE/	REAL			TCKSF	1B	/KEY1/
FDAY	1B	/DATE/	REAL			TSOLMIN	4B	/KEY1/
FHR	1B	/TIM/	REAL			TSOLSF	5B	/KEY1/
FHRE	3B	/TIM/	REAL			VPSMIN	11B	/KEY1/
FMIN	2B	/TIM/	REAL			VPSSF	12B	/KEY1/
FMINE	4B	/TIM/	REAL			WBBP	3	DUMMY-ARG
FMNTH	0B	/DATE/	REAL			WBBR	4	DUMMY-ARG
FYEAR	0B	/TIM/	REAL			WCASE	8	DUMMY-ARG
IDAYE	2507B		INTEGER	UND/*S*		WCHBR	7	DUMMY-ARG
IDAYS	2505B		INTEGER	UND/*S*		WCKT	5	DUMMY-ARG
ISTRIN	0B	/TOP/	CHAR#19			WISOL	6	DUMMY-ARG
NSTRIN	2503B		CHAR#14			WPRESS	11	DUMMY-ARG
NUM	6B	/KEY1/	INTEGER			WPRT	2	DUMMY-ARG
NW	12	DUMMY-ARG	INTEGER			WRAD	9	DUMMY-ARG
PREMIN	15B	/KEY1/	REAL			WVPS	10	DUMMY-ARG
PRESF	16B	/KEY1/	REAL			WWK	1	DUMMY-ARG
PRTMIN	0B	/KEY/	REAL			XSCALE	0B	/SCALE/

ORIGINAL PAGE IS  
OF POOR QUALITY

SUBROUTINE HEADR  
DD=-LONG/-OT,ARG= COMMON/-FIXED,CS= USER/-FIXED,DB=-TB/-SL/-ER/-ID/-PMD/-ST,-AL,PL=5000  
FTN5,I=HPLOT,L=LF.

74/860

DPT=1,ROUND= A/ S/ M/-D,-DS

FTN 5.1+642

87/04/30. 09.48.

C

```
1 SUBROUTINE HEADR
2 COMMON/TIM/FYEAR,FHR,FMIN,FHRE,FMINE
3 COMMON/DATE/FMONTH,FDAY,EMNTH,EDAY
4 COMMON/ANS,ISTRIN
5 CHARACTER ANS*1,ISTRIN*18
6 CHARACTER LSTRIN*28,ITIT1*5,ITIT2*6,ITIT3*4,ISTR5*5
7 DATA LSTRIN /'HALOE BLACKBODY PERFORMANCE'/
8 DATA ITIT1/'YEAR: '//
9 DATA ITIT2/'MONTH: '//
10 DATA ITIT3/'DAY: '//
11 DATA ISTR5/'TIME: '//
12 DATA ISTR4/'START TIME : '//
13 CALL CHARST8
```

C

```
14 CALL CHARACTER(2.5,8.0,.15,LSTRIN,0,28,.2)
15 CALL CHARACTER(2.5,7.5,.10,ITIT1,0,5,.2)
16 CALL CHARACTER(2.5,7.0,.10,ITIT2,0,6,.2)
17 CALL CHARACTER(2.5,6.5,.10,ITIT3,0,4,.2)
18 CALL CHARACTER(2.5,6.0,.10,ISTR5,0,5,.2)
19 CALL NUMBER(4.0,7.5,.10,FYEAR,0.,-1)
20 CALL NUMBER(4.0,7.0,.10,FMNTH,0.,-1)
21 CALL NUMBER(4.0,6.5,.10,FDAY,0.,-1)
22 CALL NUMBER(6.0,7.5,.10,FYEAR,0.,-1)
23 CALL NUMBER(6.0,7.0,.10,EMNTH,0.,-1)
24 CALL NUMBER(6.0,6.5,.10,EDAY,0.,-1)
25 CALL NUMBER(4.0,6.0,.10,FHR,0.,-1)
26 CALL NUMBER(4.8,6.0,.10,FMIN,0.,-1)
27 CALL NUMBER(6.0,6.0,.10,FHRE,0.,-1)
28 CALL NUMBER(6.8,6.0,.10,FMINE,0.,-1)
29 CALL CHARST1
30 RETURN
31 END
32
```

ORIGINAL PAGE IS  
OF POOR QUALITY

```

---PROCEDURES---(LD=A)
--NAME-----TYPE-----ARGS-----CLASS-----
AXES          11      SUBROUTINE
CALPLT        3      SUBROUTINE
CHARACT       7      SUBROUTINE
FLOAT         1      INTRINSIC
HEADR         0      SUBROUTINE
LEROY
LINPLT
NEWPEN
NFRAME
SUBROUTINE
SUBROUTINE
SUBROUTINE
SUBROUTINE

```

```

---STATEMENT LABELS---(LD=A)
--LABEL-ADDRESS-----PROPERTIES-----DEF
5      30B      33
15     252B     89
25     476B     146
35     720B     202
45     1136B
100    71B
200    313B
256
48
104
300    537B
400    761B
500    1173B

```

```

---ENTRY POINTS---(LD=A)
--NAME-----ADDRESS-----ARGS-----
SUB9      3B      12

```

# ---STATISTICS---

```

PROGRAM-UNIT LENGTH      2511B = 1353
CM LABELLED COMMON LENGTH      41B = 33
CM STORAGE USED      62000B = 25600
COMPILE TIME      5.764 SECONDS

```

ORIGINAL PAGE IS  
OF POOR QUALITY.

BLOCKDATA COM 74/860 OPT=1,ROUND= A/ S/ M/-D,-DS FTN 5.1+642 87/04/30. 09.48.  
DO=-LONG/-OT,ARG= COMMON/-FIXED,CS= USER/-FIXED,DB=-T8/-SB/-SL/-ER/-ID/-PMD/-ST,-AL,PL=5000  
FTN5,I=HPLOT,L=LF.

```
1 BLOCK DATA COM
2
3 C
4 COMMON/JTB/NFR,JREQ,IBAUD,HDR,IJO,TFAC,IJTB(4)
5 COMMON/SCALE/XSCALE
6 COMMON/TOP/ANS,ISTRIN
7 COMMON/TIT/IDEN1,IDEN2,IDEN3,IDEN4,IDEN5,IL1,IL2,IL3,IL4,IL5
8 COMMON/TIM/FYEAR,FHR,FMIN,FHRE,FMINE
9 COMMON/DATE/FMINTH,FDAY,EMNTH,EDAY
10 COMMON/KEY/ PRMIN,PRTSF,BBPMIN,BBPSF,BBRMIN,BBRSF
11 COMMON/KEY1/ TCKMIN,TCKSF,TCHMIN,TCHSF,TSOLMIN,TSOLSF,NUM,
12 *TCASMIN,TCASSF,VPSMIN,VPSSF,RADMIN,RADSF,PREMIN,PRESF
13 CHARACTER ANS*1,ISTRIN*18
14 DATA JREQ /2/
15 DATA PRMIN,PRTSF/980,10./
16 DATA BBPMIN,BBPSF/2,2./
17 DATA BBRMIN,BBRSF/2,5,5./
18 DATA TCKMIN,TCKSF/20,5./
19 DATA TCHMIN,TCHSF/15,5./
20 DATA TSOLMIN,TSOLSF/110,5./
21 DATA TCASMIN,TCASSF/115,5./
22 DATA VPSMIN,VPSSF/0,2./
23 DATA RADMIN,RADSF/0,2./
24 DATA PREMIN,PRESF/-0.01,4./
END
```

ORIGINAL PAGE IS  
OF POOR QUALITY.

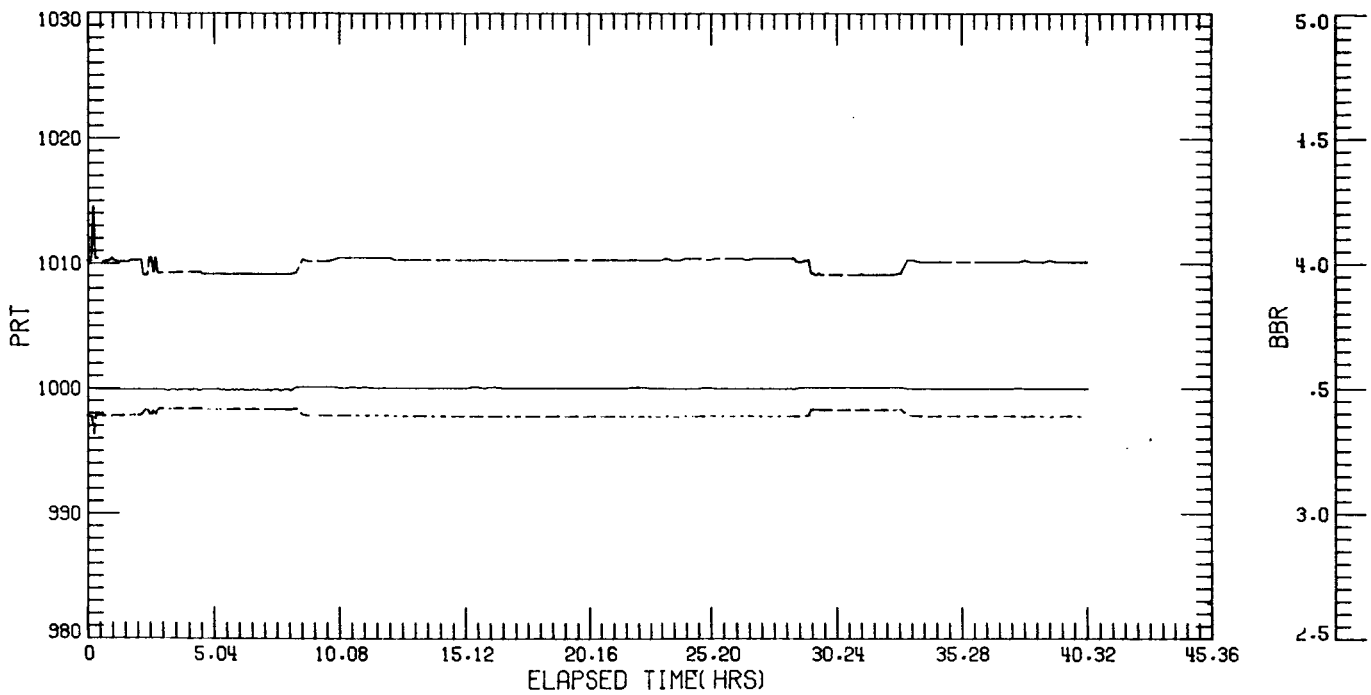
--VARIABLE MAP--(LO=A)

--NAME--ADDRESS --BLOCK--PROPERTIES--TYPE--SIZE --NAME--ADDRESS --BLOCK--PROPERTIES--

ANS	OB	/TOP/	CHAR*1	FMINE	48	/TIM/
BBPMIN	28	/KEY/	REAL	FMNTH	08	/DATE/
BBPSF	38	/KEY/	REAL	FYEAR	08	/TIM/
BBRMIN	48	/KEY/	REAL	IDEN1	08	/TIT/
BBRSF	58	/KEY/	REAL	IDEN2	18	/TIT/
EDAY	38	/DATE/	REAL	IDEN3	28	/TIT/
EMNTH	28	/DATE/	REAL	IDEN4	38	/TIT/
FDAY	18	/DATE/	REAL	IDEN5	48	/TIT/
FHR	18	/TIM/	REAL	IL1	58	/TIT/
FHRE	38	/TIM/	REAL	IL2	68	/TIT/
FMIN	28	/TIM/	REAL	IL3	78	/TIT/

# HALOE BLACKBODY PERFORMANCE

YEAR:	1985	1985
MONTH:	8	8
DAY:	13	14
TIME:	7 36	23 55

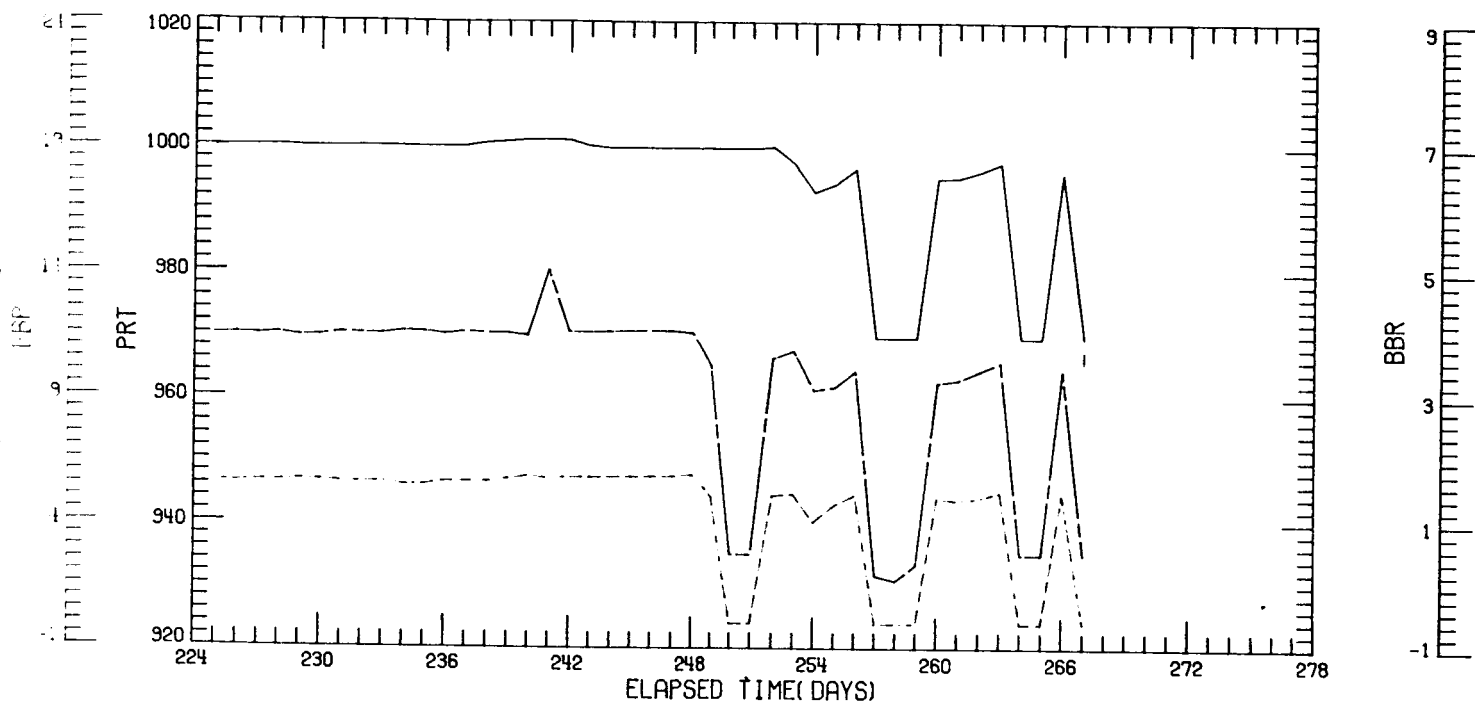




# HALOE BLACKBODY PERFORMANCE

DAILY AVERAGE

YEAR:	1985	1985
MONTH:	8	9
DAY:	13	24
TIME:	7 36	4 45



# HALOE BLACKBODY PERFORMANCE

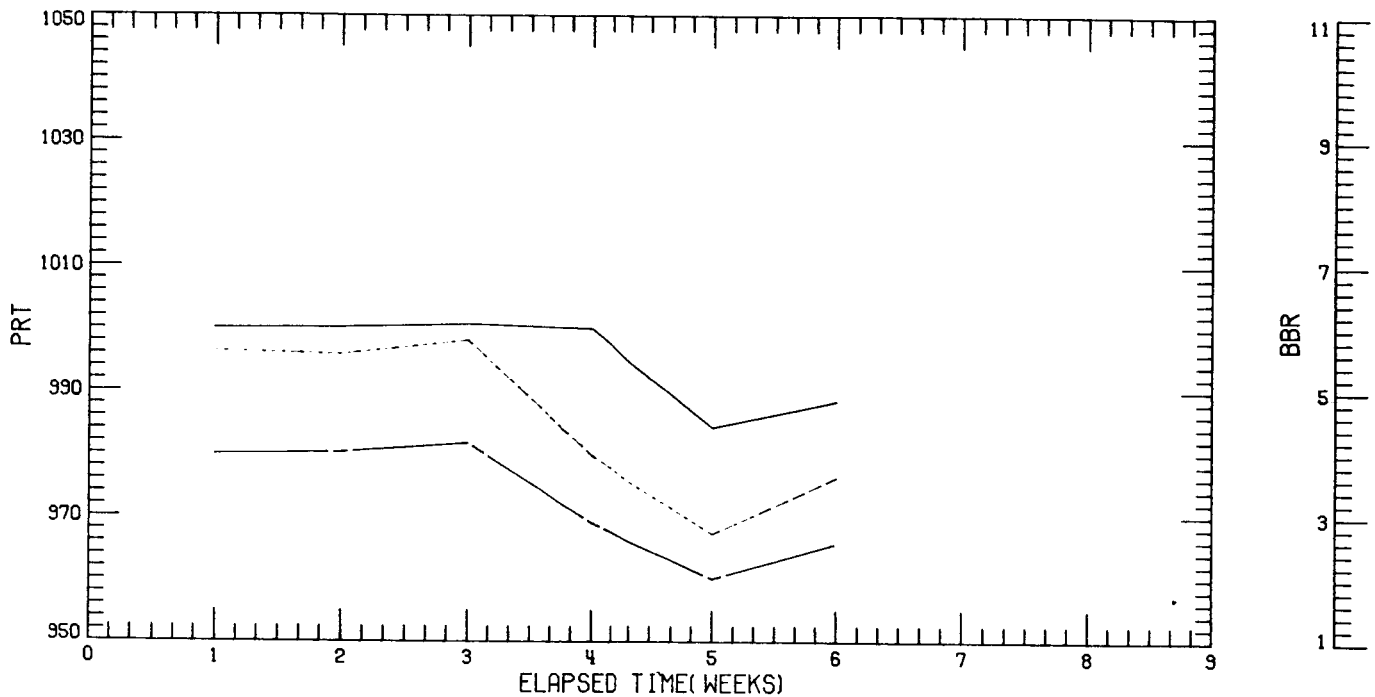
WEEKLY AVERAGE

YEAR: 1985 1985

MONTH: 8 9

DAY: 13 24

TIME: 7 36 4 45



## APPENDIX C - SPECRES

Program Name: SPECRES.PAS

Function: SPECRES is designed to acquire data from the HALOE GCETS (Gas Correlation Electronic Test Set) during the Spectral Response Test.

Description: SPECRES is written in Turbo Pascal on and for an IBM-XT or compatible. The program uses an RS232 line to communicate with the CD2A CompuDrive which drives the spectrometer during the spectral response test. SPECRES also uses a Lab Master card to acquire data from the GCETS which is in turn connected to channels of interest in the HALOE instrument.

Use: SPECRES is invoked on the IBM-XT by typing SPECRES. The program prompts the user for the run-time parameters and file names as needed. Data is saved to disk file for plotting and tabulating after each spectral run is completed.

## Listing of: SPECRES.PAS

```

1      PROGRAM SPECRES ;
3      (
4          Haloe Spectral Response data acquisition program.  This program
5          communicates with the CD2A CompuDrive to determine the wavenumber
6          setting of the spectrometer.  Each time the wavenumber changes,
7          Specres will acquire a number of data points for all the selected
8          channels.  The data is recorded on disk to be plotted and analyzed
9          immediately following a spectral response run.
11     )

14     (#U-)
15     ( RS232 INPUT/OUTPUT ROUTINES )
16     TYPE REGPACK = RECORD
17         AX,BX,CX,DX,BP,DI,SI,DS,ES,FLAGS: INTEGER ;
18     END;

20     CONST
21         SIX: BYTE = 6 ;
22         LF : BYTE = 10 ;

24     VAR
25         INSTRING : STRING[80] ;
26         RECPACK : REGPACK ;
27         AH,AL: BYTE ;
28         OLDSER,SER : INTEGER ;
29         Baud,StopBits,DataBits,PAR: Integer;
30         Message: String[80];
31         PORT1 : INTEGER ;
32         INCHAR,OUTCHAR : BYTE ;
33         INPCHAR: CHAR ABSOLUTE INCHAR ;
34         OUTPCHAR: CHAR ABSOLUTE OUTCHAR ;
35         ONLINE : BOOLEAN ;
36         printer : boolean ;
37     type
38         String19=String[19];
39     type
40         __RegisterSet=Record case Integer of
41             1: (AX,BX,CX,DX,BP,DI,SE,DS,ES,Flags: Integer);
42             2: (AL,AH,BL,BH,CL,CH,DL,DH: Byte);
43         end;
44         __ParityType=(None,Even,Odd);

46     var
47         __Regs: __RegisterSet;
48         InError,OutError: Array [1..2] of Byte;

```

## Listing of: SPECRES.PAS

```

51  (    SPECTRAL RESPONSE DATA ACQUISITION PROGRAM  )
52  TYPE
53      Filename = String[12] ;
54      Name = String[10] ;
55      Names = Array[0..4] of Name ;
56      Samples = Array[0..4] of real ;
57      descript = string[80] ;

59  LABEL STOP ;

61  CONST
62      STARTLOC : INTEGER = $710 ;
63      factor : array[0..2] of real = (1.0,10.0,100.0) ;
64      rgain: array[0..2] of integer = (1,10,100) ;
65      MAXCHANnum : INTEGER = 5 ;
66  VAR

68      PROMPT : DESCRIPT ;
69      MONTH,DAY,HR,MIN,SEC:INTEGER ;
70      bell:char;
71      NCHAN : INTEGER ;
72      ITER : INTEGER ;
73      NITER : INTEGER ;
74      NPTS : INTEGER ;
75      COUNTS : SAMPLES ;
76      i,j,k : integer ;
77      IT: INTEGER ;
78      CTRLBYTE : BYTE ;
79      STATBYTE : BYTE ;
80      Inch : integer ;
81      INPCH : ARRAY[0..15] OF INTEGER ;
82      IND : INTEGER ;
83      HIGH : BYTE ;
84      LOW : BYTE ;
85      val : real ;
86      ref : real ;
87      ICHAN : INTEGER ;
88      CHAN : ARRAY[0..5,0..1000] OF REAL ;
89      gain : array[0..15] of byte ;
90      igain : byte ;
91      sum,sumx2,mean,minx,maxx,std,nopts : samples ;
92      tsum,tsumx2,tmean,tminx,tmaxx,tstd,tnopts:samples;
93      NAM : NAMES;
94      F1 : TEXT ;
95      FNAME : FILENAME ;
96      PLOTS : TEXT ;
97      FNAME : FILENAME ;
98      IOerror : integer ;
99      answer : string[1] ;
100     WAVEL : REAL ;
101     WAVELENGTH : real ;      ( USED FOR WAVENUMBER )

```

Listing of: SPECRES.PAS

```

102      DELTA : REAL ;           { STEP SIZE }
103      DWELL , STEPS: INTEGER ;
104      denom : real ;
105      RADICAL : REAL ;
106      descrip : descript ;
107      RSINT : ARRAY[0..1] OF INTEGER ABSOLUTE $0000:$0030 ;
108      OLDINT : ARRAY[0..1] OF INTEGER ;
109      BUF,PTR,BASE : INTEGER ;
110      BUFOUT,BUFIN : INTEGER ;
111      Function Binary(V: Integer): String19;

113      var
114          I: Integer;
115          B: Array [0..3] of String[4];

117      begin
118          For I:=0 To 15 do
119              if (V and (1 Shl (15-I)))<>0 then B[I Div 4][I Mod 4+1]:='1'
120              else B[I Div 4][I Mod 4+1]:='0';
121              For I:=0 To 3 do B[I][0]:=Chr(4);
122              Binary:=B[0]+' '+B[1]+' '+B[2]+' '+B[3];
123      end;

126      function KEYIN : INTEGER ;
127      begin
128          with recpack do
129              begin
130                  ah := 6 ;
131                  al := 0;
132                  ax := ah shl 8 + al ;
133                  dx := $ff ;
134                  intr($21,recpack);
135                  al := ax and $ff ;
136                  KEYIN := al ;
137              END;
138      end;

145      FUNCTION CHANNEL (CHANNUM: INTEGER): REAL;
146      begin
147          ctrlbyte := 128 or gain[CHANNUM] ;
148          port[startloc+4] := ctrlbyte ;
149          PORT[STARTLOC+5] := INPCH[CHANNUM] ;
150          PORT[STARTLOC+6] := 0 ;
151          WHILE PORT[STARTLOC+4] and 128 = 0 DO
152              BEGIN

```

## Listing of: SPECRES.PAS

```

153         statbyte := port[startloc+4] ;
154     END;
155     LOW := PORT[STARTLOC+5] ;
156     HIGH := PORT[STARTLOC+6] ;
157     VAL := high*256.0 + low ;
158     if VAL > 32767.0 then CHANNEL := VAL -65536.0
159     ELSE CHANNEL := VAL ;

161 end;

163 PROCEDURE SETGAINS ;
164 BEGIN
165     { determine best gain value for each channel }
166     INCH := 0 ;
167     repeat
168         igain := 0 ;
169         VAL := CHANNEL(15);
170         VAL := CHANNEL(INCH) ;
171         counts[inch] := val ;
172         if (abs(val)<200.0)then
173             begin
174                 igain := 1 ;
175                 if (abs(val)<20.0)then
176                     begin
177                         igain := 2 ;
178                     end;
179                 end;
180                 gain[inch] := igain ;
181                 inch := inch +1 ;
182             until inch = nchan ;

184     END;
185     { read a burst of data }
186     procedure readburst ;
187     BEGIN
188         { initialize stats and gains }
189         for ichan := 0 to NCHAN - 1 do
190             begin
191                 sum[ichan] := 0.0 ;
192                 sumx2[ichan] := 0.0 ;
193                 minx[ichan] := 1.0e+33 ;
194                 maxx[ichan] := -1.0E+33 ;
195                 nopts[ichan] := 0 ;
196                 gain[ichan] := 0 ;
197             end;

199     SETGAINS ;           { DETERMINE BEST GAIN SETTING FOR EACH CHANNEL }

201     { acquire data }

203     ind := 0 ;

```

## Listing of: SPECRES.PAS

```

204  repeat
205      INCH := 0 ;
206      repeat
207          NOPTS[INCH] := NOPTS[INCH]+1 ;
208          VAL := CHANNEL(15);
209          VAL := CHANNEL(INCH) ;      { read ground, REF , THEN CHANNEL }
210          { IF(REF<>0.0) THEN
211              val := val/ref
212          ELSE
213              WRITELN(' DIVIDE BY ZERO REF VOLTS');
214          }
215          val := val/(204.75*factor[ gain[ inch ] ]) ;
216          sum[ inch ] := sum[ inch ] + val ;
217          sumx2[ inch ] := sumx2[ inch ] + val*val ;
218          if val < minx[ inch ] then minx[ inch ] := val ;
219          if val > maxx[ inch ] then maxx[ inch ] := val ;
220          inch := inch + 1 ;
221      until inch = nchan ;
222      IND := IND + 1;
223  UNTIL IND = ITER ; { ITER IS NUMBER ITERATIONS PER BURST }
224  end ;

228  Procedure MAKEfile( VAR FL:TEXT; PROMPT:DESCRIPT ;
229                      VAR FNAME:FILENAME; var ioerror:integer) ;
230  LABEL AGIN ;
231  begin
232      {#I-}      { turn off i/o error checking }
233  AGIN:      Writeln(PROMPT) ;
234      Readln(FNAME) ;
235      Assign(f1,FNAME);
236      Reset(f1); { try to rewind the file }
237      IOerror := IOresult ;
238      if (IOerror <> 0) then { an error will occur if it doesn't exist }
239      begin
240          Rewrite(F1) ;      { try to create the file }
241          IOerror := IOresult ;
242          if (IOerror <> 0) then writeln(' error in creating file: ',IOerror:5);
243      end
244      else
245      begin
246          writeln(' FILE ALREADY EXISTS, DO YOU WANT TO OVERWRITE IT? (Y/N)');
247          READLN(ANSWER);
248          IF (UPCASE(ANSWER)= 'Y' ) THEN
249              BEGIN
250                  CLOSE(FL);
251                  GOTO AGIN ;
252              END;
253      end;
254  end;

```



Listing of: SPECRES.PAS

```

257  FUNCTION BCD2DEC(X:INTEGER) : INTEGER ;
258  BEGIN
259      BCD2DEC := (X DIV 16) * 10 + (X MOD 16) ;
260  END ;

262  PROCEDURE TIME(VAR MONTH,DAY,HR,MIN,SEC:INTEGER) ;
263  CONST TIMEBASE = 893 ;
264  BEGIN
265      PORT[TIMEBASE] := 2 ;    { SELECT SECONDS REGISTER }
266      SEC := BCD2DEC(PORT[TIMEBASE+2]);
267      PORT[TIMEBASE] := 3 ;    { SELECT MINUTES REGISTER }
268      MIN := BCD2DEC(PORT[TIMEBASE+2]);
269      PORT[TIMEBASE] := 4 ;    { SELECT HOURS REGISTER }
270      HR := BCD2DEC(PORT[TIMEBASE+2]);
271      PORT[TIMEBASE] := 6 ;    { SELECT DAY OF MONTH }
272      DAY := BCD2DEC(PORT[TIMEBASE+2]);
273      PORT[TIMEBASE] := 7 ;    { SELECT MONTH REGISTER }
274      MONTH := BCD2DEC(PORT[TIMEBASE+2]);
275  END;

277  Procedure Selectchannels ;
278  Var i: integer ;
279  Begin
280      WRITELN(' ENTER THE NUMBER OF CHANNELS');
281      READLN(NCHAN);
282      for i:= 0 to NCHAN - 1 do
283          begin
284              writeln(' Enter description of channel# ',i:5);
285              readln(NAM[I]);
286              WRITELN(' ENTER PLUG POSITION# FOR THIS CHANNEL');
287              READLN(INPCH[I]);
288          end;
289      WRITELN(' BE SURE THAT THE GROUND (SHORTING) PLUG IS IN POSITION 15');
290      INPCH[15] := 15 ;
291  end;

293  PROCEDURE ASCIN ; EXTERNAL 'ASCIN.COM' ;

295  PROCEDURE ASCINIT ;
296  BEGIN
297      BASE := OFS(ASCIN) ;
298      PTR := BASE + $2D ;
299      BUF :=BASE + $2F ;
300      MEMWLCSEG:BASE+$10] := PTR ;
301      MEMWLCSEG:BASE+$14] := BUF ;
302      MEMWLCSEG:BASE+$21] := PTR ;

304  END;

```

Listing of: SPECRES.PAS

```

307  procedure ASCII_ENABLE ;
308  BEGIN
309      PORT[$3FC] := $0B ;
310      PORT[$21] := PORT[$21] AND $EF ;
311      PORT[$3F9] := 1 ;
312  END;

315  FUNCTION DATA_AVAIL : BOOLEAN ;
316  BEGIN
317      DATA_AVAIL := TRUE ;
318      BUFIN := MEMWICSEG:PTR ;
319      IF BUFIN = BUFOUT THEN DATA_AVAIL := FALSE ;

321  END;

323  { Beginning of Main Program -----}
324  Begin

326  OLDINT[0] := RSINT[0] ;
327  OLDINT[1] := RSINT[1] ;
328  ASCINIT ;
329  RSINT[0] := OFS(ASCIN);
330  RSINT[1] := CSEG ;
331  BUFOUT := 0 ;
332  ASCII_ENABLE ;
333  REPEAT
334      OUTCHAR := KEYIN ;
335      IF OUTCHAR <> 0 THEN
336      BEGIN
337          { REPEAT
338              UNTIL ((PORT[$3FD] AND $20) <> 0 ) ;}
339              PORT[$3F8] := OUTCHAR ;
340          END ;
341          WHILE DATA_AVAIL DO
342          BEGIN
343              INCHAR := MEMICSEG:BUF+BUFOUT ;
344              BUFOUT := BUFOUT+ 1;
345              IF BUFOUT > 255 THEN BUFOUT := 0 ;
346              CASE INCHAR OF
347                  32..128,10,13: WRITE(INFCHAR);
348                  5: BEGIN
349                      { REPEAT
350                          UNTIL ((PORT[$3FD] AND $20) <> 0 ) ;}
351                          PORT[$3F8] := 6 ;
352                      END;
353                      END ;
354                  END;
355              UNTIL INCHAR = 26 ;

```

```
357  WRITELN(' SPECTRAL RESPONSE DATA ACQUISITION PROGRAM' );
358  WRITELN ;
359  WRITELN(' written by William L. Edmonds ' );
360  writeln;
361  writeln;
362  writeln;
363  bell := chr(#07);
364  PROMPT := ' ENTER FILE NAME FOR SPECTRAL RESPONSE DATA (ALL PTS) ' ;
365  MAKEFILE(FL,PROMPT,FNAME,IOERROR) ;
366  IF(IOERROR <> 0 ) THEN GOTO STOP ;
367  PROMPT := ' ENTER FILE NAME FOR PLOT FILE ' ;
368  MAKEFILE(PLOTS,PROMPT,PNAME,IOERROR);
369  Selectchannels ;
370  WRITELN(' ENTER TOTAL NUMBER OF DATA POINTS FOR EACH WAVELENGTH');
371  READLN(NPTS);
372  ITER := 10 ;
373  NITER := NPTS DIV ITER ;

375      writeln(' Enter description of this run (80 chars)');
376      readln(descrip);
377      writeln(' Enter START WAVENUMBER (real number with decimal)');
378      readln(WAVELENGTH);
379      WRITELN(' ENTER DELTA WAVENUMBER (REAL NUMBER )');
380      READLN(DELTA);
381      WRITELN(' ENTER NUMBER OF STEPS (INTEGER)');
382      READLN(STEPS);
383      WRITELN(' ENTER DWELL TIME IN SECONDS (INTEGER)');
384      READLN(DWELL);
385      writeln(' Type B when ready to start taking data ' ) ;
386      writeln(' OR enter Q to quit' ) ;
387      readln(answer);
388      IF(UPCASE(ANSWER)<>'G') THEN GOTO STOP;
389      WAVEL := WAVELENGTH ;
390      writeln(fl,descrip);
391  FOR j:= 1 TO STEPS DO      { wavenumber loop }
392  BEGIN
393      if (UPCASE(answer) <> 'G' ) then goto stop ;
394      WRITELN(' WAVELENGTH = ',WAVEL:10:2);
395      WRITELN(LST,' WAVELENGTH = ',WAVEL:10:2);
396      TIME(MONTH,DAY,HR,MIN,SEC);
397      WRITELN(fl,MONTH:2,'/',DAY:2,'/86 ',HR:2,':',MIN:2,':',SEC:2);
398      writeln(fl,
399          'parameter      minimum      maximum      mean      std dev      num pts');
400      WRITELN(lst,
401          MONTH:2,'/',DAY:2,'/86 ',HR:2,':',MIN:2,':',SEC:2);
402      writeln(lst,
403          'parameter      minimum      maximum      mean      std dev      num pts');
404      WRITELN(MONTH:2,
405          '/',DAY:2,'/86 ',HR:2,':',MIN:2,':',SEC:2);
406      writeln(
407          'parameter      minimum      maximum      mean      std dev      num pts');
```

## Listing of: SPECRES.PAS

```

408      WRITELN(FL, ' WAVELENGTH = ', WAVELENGTH);
409      for ind := 0 to nchan - 1 do
410      BEGIN
411          tmean[ind] := 0.;
412          tsum[ind] := 0.;
413          tsumx2[ind] := 0.;
414          tminx[ind] := 1.0e+33;
415          tmaxx[ind] := -1.0e+33;
416          tnopts[ind] := 0.;
417          tstd[ind] := 0.;
418      end;
419      ( readburst ; read each channel to initialize process )
420      for k := 1 to niter do
421      begin
422          ( readburst ; )
423          for ind := 0 to NCHAN - 1 do
424          begin
425              mean[ind] := sum[ind]/nopts[ind] ;
426              RADICAL := (nopts[ind]*sumx2[ind]-sum[ind]*sum[ind])/
427                          ((nopts[ind]-1)*nopts[ind]) ;
428              tsum[ind] := tsum[ind]+sum[ind];
429              tsumx2[ind] := tsumx2[ind]+sumx2[ind];
430              if (minx[ind]<tminx[ind]) then tminx[ind] := minx[ind];
431              if (maxx[ind]>tmaxx[ind]) then tmaxx[ind] := maxx[ind];
432              tnopts[ind] := tnopts[ind]+nopts[ind];
433              IF (RADICAL>0.0) THEN
434              BEGIN
435                  STD[IND] := SQRT(RADICAL) ;
436              END
437              ELSE
438              BEGIN
439                  STD[IND] := 0.0 ;
440              END;
441              (
442                  writeln(NAM[ind]:10, mean[ind]:8:4, std[ind]:10:4
443                      , factor[gain[ind]]:5:1);
444              )
445              writeln(FL, nam[IND]:10, minX[IND]:10:5, maxx[IND]:10:5,
446                  mean[IND]:10:5, std[IND]:10:5, NOPTS[IND]:10:0);
447              (
448                  writeln(1st, nam[IND]:10, minX[IND]:10:5, maxx[IND]:10:5,
449                      mean[IND]:10:5, std[IND]:10:5, NOPTS[IND]:10:0);
450              )
451          end; ( of for loop )
452          writeln(f1);
453      end;
454
455      for ind := 0 to nchan - 1 do
456      begin
457          tmean[ind] := tsum[ind]/tnopts[ind] ;
458          radical := 0.0 ;
459          denom := ((tnopts[ind]-1)*tnopts[ind]);
460          if (denom<>0.0) then
461              radical := (tnopts[ind]*tsumx2[ind]-tsum[ind]*tsum[ind])
462                          / denom;

```

Listing of: SPECRES.PAS

```
459         if(radical>0.0)then
460         begin
461             tstd[ind] := sqrt(radical);
462         end
463         else
464         begin
465             tstd[ind] := 0. ;
466         end;
467         writeln(f1,nam[ind]:10,tminx[ind]:10:5,tmaxx[ind]:10:5,
468             tmean[ind]:10:5,tstd[ind]:10:5,tnopts[ind]:10:5);
469         writeln(lst,nam[ind]:10,tminx[ind]:10:5,tmaxx[ind]:10:5,
470             tmean[ind]:10:5,tstd[ind]:10:5,tnopts[ind]:10:5);
471         writeln(nam[ind]:10,tminx[ind]:10:5,tmaxx[ind]:10:5,
472             tmean[ind]:10:5,tstd[ind]:10:5,tnopts[ind]:10:5);
473     end;
474     WAVEL := WAVELENGTH + j*DELTA ;
475     WRITELN(bell,
476         'ENTER G WHEN READY TO TAKE DATA FOR WAVELENGTH = '
477         ,WAVEL:10:2);
478     READLN(ANSWER);
479 END;
480 stop:
481     WRITELN(FL);
482     close(f1) ;
483 END.
```

## APPENDIX D - SPECPLT

Program Name: SPECPLT.PAS

Function: SPECPLT is designed to plot HALOE spectral response data on an HP pen plotter.

Description: SPECPLT is written in Turbo Pascal for an IBM-XT or compatible.

Use: After each spectral response run is made, it is essential to plot the data to determine the quality of the data and whether or not an additional run under the same conditions is necessary. SPECPLT gives the capability of plotting the data quickly, allowing several parameters to be plotted in different colors on the same graph.

Listing of: A:SPECPLT.PAS

```

1  (*****
2  (*****
3  (*
4  (*          TURBO PASCAL PLOT PROGRAM for Spectral
5  (*          Response using IEEE 488 BUS DRIVER
6  (*
7  (*
8  (*****
9  (*****
10 Program Specplt ;
11 type
12     filename = string[127] ;
13     name = string[10] ;
14     names = array[0..16] of name ;
15     cmd = string[127] ;
16     VALUE = STRING[10] ;
17     vax = string[80] ;
18     flg = integer ;
19     bad = integer ;
20     INTS = ARRAY[0..10] OF INTEGER ;
21     ANTS = ARRAY[0..21] OF BYTE ;
22     param = array[1..200] of real ;
23     STRG = STRING[40] ;
24 CONST ZERO : STRING[3] = ' 0 ' ;
25     MINEQ : STRING[6] = 'MIN = ' ;
26     MAXEQ : STRING[6] = 'MAX = ' ;
27     MINIMUM : REAL = 1.0E+33 ;
28     MAXIMUM : REAL = -1.0E+33 ;
29     ET : BYTE = 3 ;
31 Label TOP,NEWPLOT,theEnd ;
32 var
33     ETX : CHAR ABSOLUTE ET ;
34     PENPOS : VAX ;
35     LAB : STRG ;
36     ANSWER : CHAR ;
37     nparam,CHAN : integer ;
38     params : array[1..16] of param ;
39     PARVAL : string[10] ;
40     parnam : ARRAY[0..16] OF name ;
41     PARNAME : NAME ;
42     waveleng : param ;
43     WAVEVAL : string[10] ;
44     parmin,parmax : array[1..16] of real ;
45     wavemin,wavemax : real ;
46     title : STRG ; { title of plot can be up to 40 characters }
47     XLAB,YLAB,DIR:VALUE ;
48     date,datetime : value ; { 10 character strings for date and time }
49     XCOORD,YCOORD : REAL ;
50     I,J,npt:INTEGER ;

```

Listing of: A:SPECPLT.PAS

```

51     X,Y: VALUE ;
52     MINX,MINY,MAXX,MAXY : REAL ;
53     XSF,YSF,XOF,YOF : REAL ; { X&Y SCALE FACTORS AND OFFSETS }
54     XTIC : VALUE ;
55     XPOS : REAL ;
56     XDIV,YDIV : INTEGER ;
57     XDEL,YDEL,ydelta : REAL ;
58     syscon:cmd;
59     f:flg;
60     b:bad;
61     v:vax;
62     RX,RY:REAL ;
63     c:cmd;
64     IANS: CHAR ;
65     NUMS: INTS ABSOLUTE V ;
66     BYTES : ANTS ABSOLUTE V ;
67     TEMP : BYTE ;
68     specfile : text ;
69     specfilename : filename ;
70     ioerror : integer ;
71     PEN : CHAR ;

73     Procedure Openfile(var FL:TEXT;var FNAME:FILENAME;var ioerror:integer) ;
74     LABEL AGIN ;
75     begin
76     {$I-}          { turn off i/o error checking }
77     AGIN:          Writeln(' Enter plot data file name ' );
78                   Readln(FNAME) ;
79                   Assign(fl,FNAME);
80                   Reset(fl);
81                   IOerror := IOresult ;
82                   if(IOerror <> 0) then
83                   begin
84                       writeln(' File : ',fname,' does not exist! ');
85                       writeln(' DO YOU WANT TO TRY AGAIN? (Y/N)');
86                       READLN(ANSWER);
87                       IF (UPCASE(ANSWER) = 'Y' ) THEN
88                           goto agin ;
89                   end
90                   else
91                   begin
92                       writeln(' OPENING FILE: ',FNAME);
93                   end;
94     end;

96     procedure ReadInData ( var ioerror : integer ) ;
97     VAR PRINT : BOOLEAN ;
98     LABEL FINIS ;
99     begin
100         WRITELN(' DO YOU WANT TO PRINT THE DATA?');
101         READLN(ANSWER);

```



Listing of: A:SPECPLT.PAS

```

102      IF UPCASE(ANSWER) = 'Y' THEN PRINT := TRUE ELSE PRINT := FALSE ;
103      readln(specfile,title);
104      writeln(' title : ', title ) ;
105      readln(specfile,date,datetime);
106      writeln(' date and time : ',date,datetime);
107      readln(specfile,nparam);
108      writeln(' number of parameters = ',nparam:5);
109      READ(SPECFILE,PARNAM[0]);
110      WAVEMIN := MINIMUM ;
111      WAVEMAX := MAXIMUM ;
112      IF PRINT THEN
113      BEGIN
114          WRITELN(LST,TITLE);
115          WRITELN(LST,DATE,DATETIME);
116          WRITELN(LST,' NUMBER OF PARAMETERS = ',NPARAM);
117          WRITE(LST,PARNAM[0]);
118      END;
119      for i := 1 to nparam do
120      begin
121          read(specfile,parnam[i]);
122          IF PRINT THEN WRITE(LST,PARNAM[i]);
123          PARMIN[i] := MINIMUM ;
124          PARAMAX[i] := MAXIMUM ;
125      end;
126      IF PRINT THEN WRITELN(LST);
127      npt := 0 ;
128      repeat
129          npt := npt + 1 ;
130          read(specfile,waveLENG[npt]);
131          IF EOF(SPECFILE) THEN GOTO FINIS ;
132          IF PRINT THEN WRITE(LST,WAVELENG[NPT]:10:2);
133          for j:= 1 to nparam do
134          begin
135              read(specfile,paramS[j,npt]);
136              IF EOF(SPECFILE) THEN GOTO FINIS ;
137              IF PRINT THEN WRITE(LST,PARAMS[J,NPT]:10:5);
138          end;
139          IF PRINT THEN WRITELN(LST);
140      until eof(specfile) ;
141      FINIS: NPT := NPT-1 ;
142      FOR I := 1 TO NPT DO
143      BEGIN
144          IF WAVELENG[i] < WAVEMIN THEN WAVEMIN := WAVELENG[i] ;
145          IF WAVELENG[i] > WAVEMAX THEN WAVEMAX := WAVELENG[i] ;
146          FOR J := 1 TO NPARAM DO
147          BEGIN
148              IF PARAMS[J,i] < PARMIN[J] THEN PARMIN[J] := PARAMS[J,i] ;
149              IF PARAMS[J,i] > PARAMAX[J] THEN PARAMAX[J] := PARAMS[J,i] ;
150          END ;
151      END;
152      end;

```

Listing of: A:SPECFLT.PAS

```

154  procedure IE488 ( VAR c:cmd;
155                      VAR v:vax;
156                      VAR f:flg;
157                      VAR b:bad ); external 'IE488.COM';

159  PROCEDURE LABELIT(VAR LAB:STRG; VAR X,Y, DIRECTION: VALUE);
160  BEGIN
161      V:='DI ' + DIRECTION + ' ; ' ;
162      IE488(C,V,F,B);
163      V := 'PU PA ' + X + Y + ' ; ' ;
164      IE488(C,V,F,B);
165      V := 'LB ' + LAB + ETX ;
166      IE488(C,V,F,B);
167  END;

171  PROCEDURE INITIEEE ;
172  BEGIN

174      f := 1;
175      b := 0;
176      syscon := 'SYSCON MAD=3, CIC=1, NOB=1, BAO=&H200';
177      v := ' ';
178      IE488(syscon,v,f,b);
179      if f<> 0 then
180          writeln('RETURNED FROM IE488 SYSCON PROCEDURE flg = ', f);
181      F:= 0;
182      B:= 0 ;
183      C := 'TIMEOUT' ;
184      V := chr(1) ;
185      IE488(C,V,F,B);
186      if f<>0 then
187          WRITELN(' TIMEOUT PROC RETURN WITH FLAG =',F);
188      C:= 'OUTPUT 5[##]' ;
189  END;

191  PROCEDURE INITPLOT ;
192  BEGIN

194      V := 'DF IN PS 4 IF 0,0,9865,7462;' ;
195      IE488(C,V,F,B);
196      V := ' SC -20,100,-10,110 ;';
197      IE488(C,V,F,B);
198      if f<>0 then
199          WRITELN(' INITIALIZED PLOTTER, FLAG = ',F);
200          WRITELN(' WHAT PEN NUMBER DO YOU PREFER?');
201          READLN(PEN);
202          V:= 'SP ' + PEN + ' ; ' ;
203          IE488(C,V,F,B);

```

Listing of: A:SPECPLT.PAS

```

204     V := 'PA 0,0,PD 100,0,100,100,0,100,0,0 ;' ;
205     IE488(C,V,F,B);
206     V:= ' PU 0,0 ;' ;
207     IE488(C,V,F,B);

209  END;

211  PROCEDURE AXES;
212  BEGIN
213      XDEL := 100.0/XDIV ;
214      YDEL := 100.0/YDIV ;
215      V:= ' ' ;
216      FOR I:= 1 TO XDIV DO
217          BEGIN
218              XPOS := I*XDEL ;
219              STR(XPOS:8:4,XTIC);
220              V := 'PA ' + XTIC + ',' + ZERO + ';' + 'XT;' ;
221              IE488(C,V,F,B);
222          END;
223      FOR I := 1 TO YDIV DO
224          BEGIN
225              XPOS := I * YDEL ;
226              STR(XPOS:8:4,XTIC);
227              V := 'PA ' + ZERO + ',' + XTIC + ';' + 'YT;' ;
228              IE488(C,V,F,B);
229          END;
230      V := 'PU PA 0,0 ;' ;
231      IF F<>0 THEN WRITELN(' ERROR IN AXES = ',F);

233  END;
234  procedure plotline ;
235  BEGIN
236      I := 1 ;
237      XCOORD := (WAVELENG[I]-XOF)*XSF ;
238      YCOORD := (PARAMS[CHAN,I]-YOF)*YSF ;
239      STR(XCOORD:10:2,WAVEVAL);
240      STR(YCOORD:10:2,PARVAL);
241      penpos := 'PU ' ;
242      V := penpos + WAVEVAL + ',' + PARVAL + ';' ;
243      IE488(C,V,F,B) ;
244      PENPOS := 'PD ' ;
245      FOR I := 1 TO NPT DO
246          BEGIN
247              XCOORD := (WAVELENG[I]-XOF)*XSF ;
248              YCOORD := (PARAMS[CHAN,I]-YOF)*YSF ;
249              STR(XCOORD:10:2,WAVEVAL);
250              STR(YCOORD:10:2,PARVAL);
251              V := penpos + WAVEVAL + ',' + PARVAL + ';' ;
252              IE488(C,V,F,B) ;
253      END ;

```

Listing of: A:SPECPLT.PAS

```

255  END;

257  PROCEDURE SETSCALES ;
258  BEGIN

260      WRITELN(' CURRENT WAVENUMBER MIN AND MAX ARE: ',WAVEMIN:10:2,
261      WAVEMAX:10:2);
262      WRITELN(' CURRENT MIN AND MAX FOR ',PARNAM[CHAN],': ',
263      PARMIN[CHAN]:10,' ',PARMAX[CHAN]:10);
264      writeln(' DO YOU WANT TO ADJUST THESE? (Y/N) ');
265      READLN(ANSWER);
266      IF (UPCASE(ANSWER) = 'Y') THEN
267  REPEAT
268      WRITELN(' ENTER WAVENUMBER MINIMUM: ');
269      READLN(WAVEMIN);
270      WRITELN(' ENTER WAVENUMBER MAXIMUM: ');
271      READLN(WAVEMAX);

273      WRITELN(' ENTER MIN FOR: ',PARNAM[CHAN]);
274      READLN(PARMIN[CHAN]);
275      WRITELN(' ENTER MAX FOR: ',PARNAM[CHAN]);
276      READLN(PARMAX[CHAN]);
277      WRITELN(' MIN AND MAX WAVENUMBERS: ',WAVEMIN:10:2,WAVEMAX:10:2);
278      WRITELN(' MIN AND MAX FOR ',PARNAM[CHAN],PARMIN[CHAN]:10,
279      ' ',PARMAX[CHAN]:10);
280      WRITELN(' ARE THESE VALUES OK? (Y/N) ');
281      READLN(ANSWER);
282  UNTIL UPCASE(ANSWER) = 'Y' ;
283  XDEL := WAVEMAX-WAVEMIN ;
284  YDEL := PARMAX[CHAN]-PARMIN[CHAN] ; ;
285  XSF := 100.0/XDEL ;
286  YSF := 100.0/YDEL ;
287  XOF := WAVEMIN ;
288  YOF := PARMIN[CHAN] ;
289  ydelta := ydel ;
290  END ;

293  PROCEDURE YLABEL (pmin,pmax:real;pnam:name) ;
294  BEGIN

296      V := 'PU PA O O ' ;
297      IE488(C,V,F,B);
298      YLAB := ' O ' ;
299      STR(Pmin:10,LAB);
300      LAB := MINEQ + LAB ;
301      DIR := ' O 1 ' ;
302      LABELIT(LAB,XLAB,YLAB,DIR);

304      YLAB := '40 ' ;
305      LAB := Pnam ;

```

Listing of: A:SPECPLT.PAS

```

306  DIR := '0 1 ' ;
307  LABELIT(LAB,XLAB,YLAB,DIR);

309  YLAB := ' 70 ' ;
310  STR(Pmax:10,LAB);
311  LAB := MAXEQ + LAB ;
312  LABELIT(LAB,XLAB,YLAB,DIR);

314  END;

316  {----- S P E C P L T  MAIN PROGRAM -----}

318  BEGIN
319  INITIEEE ;           { INITIALIZE IEEE BUS FOR PLOTTING }

321  TOP:      OPENFILE(specfile,specfilename,ioerror);

323  if ioerror <> 0 then goto theEnd ;
324  Readindata(ioerror) ;
325  if ioerror <> 0 then goto theEnd ;

327  NEWPLOT:

329  XDIV := 10 ;
330  YDIV := 10 ;

332  FOR I:= 1 TO NPARAM DO
333  WRITELN('CHANNEL# ',I:5,PARNAM[I]:12) ;
334  WRITELN(' ENTER CHANNEL # TO PLOT AGAINST WAVELENGTH');
335  READLN(CHAN);
336  SETSCALES ;

338  INITPLOT ;
339  AXES ;
340  XLAB := ZERO ;
341  YLAB := '100 ' ;
342  DIR := ' 1 0 ' ;
343  LABELIT(TITLE,XLAB,YLAB,DIR);
344  XLAB := ' 50 ' ;
345  LAB := DATE + ' ' + DATIME ;

347  LABELIT(LAB,XLAB,YLAB,DIR);

349  PLOTLINE;

352  XLAB := ' 0 ' ;
353  YLAB := '-5 ' ;
354  STR(WAVEMIN:8:2,LAB);
355  LAB := MINEQ + LAB ;
356  LABELIT(LAB,XLAB,YLAB,DIR);

```

Listing of: A:SPECPLT.PAS

```

358  XLAB := '40 ' ;
359  YLAB := '-5 ' ;
360  LAB := PARNAM[0] ;
361  LABELIT(LAB,XLAB,YLAB,DIR);

363  XLAB := ' 70 ' ;
364  STR(WAVEMAX:8:2,LAB);
365  LAB := MAXEQ + LAB ;
366  LABELIT(LAB,XLAB,YLAB,DIR);

368  XLAB := '-5 ' ;
369  YLABEL(parmin[chan],parmax[chan],parnam[chan]) ;

371  WRITELN(' DO YOU WANT TO PLOT ANOTHER CHAN ON SAME PLOT? (Y/N) ');
372  READLN(ANSWER);
373  IF UPCASE(ANSWER) = 'Y' THEN
374  BEGIN
375      WRITELN(' WHAT PEN NUMBER DO YOU PREFER? ');
376      READLN(PEN);
377      V:= 'SP ' + PEN + ' ';
378      IE488(C,V,F,B);
379      XLAB := '-10 ' ;
380      FOR I:= 1 TO NPARAM DO
381          WRITELN(' CHANNEL# ',I:5,PARNAM[I]:12) ;
382      WRITELN(' ENTER CHANNEL # TO PLOT AGAINST WAVELENGTH ');
383      READLN(CHAN);
384      WRITELN(' DO YOU WANT TO USE THE SAME SCALE-FACTOR (Y/N) ');
385      READLN(ANSWER);
386      IF UPCASE(ANSWER) = 'N' THEN
387          SETSCALES
388      else
389          YOF := 0.0 ;

391      YLABEL(0.0,ydelta,parnam[chan]) ;
392      PLOTLINE ;

394  END;

398  v := ' SP 0 ' ;
399  IE488(C,V,F,B);
400  WRITELN(' DO YOU WANT TO CONTINUE? (Y/N) ');
401  READLN(ANSWER);
402  IF (UPCASE(ANSWER)='N') THEN
403      GOTO THEEND
404  ELSE
405      BEGIN
406          WRITELN(' SAME FILE?(Y/N) ');
407          READLN(ANSWER);

```

Listing of: A:SPECPLT.PAS

```
408      IF (UPCASE (ANSWER)='Y') THEN GOTO NEWPLOT;  
409      CLOSE (SPECFILE);  
410      GOTO TOP;  
411  END;  
412  THEEND:  
413  CLOSE (SPECFILE);  
414  END.
```

## APPENDIX E - MONITOR

Program Name: MONITOR.PAS

Function: MONITOR is designed to acquire HALOE major frames of data and to limit check the data before displaying it on a color monitor in color coded form. MONITOR will also archive data to disk for off-line processing.

Description: MONITOR is a Turbo Pascal program written on an IBM-XT.

Use: MONITOR will be used to limit check, display and archive HALOE major frames of data during refurb testing and UARS I & T (Upper Atmosphere Research Satellite Integration and Testing). It will be part of an overall quick-look system for HALOE.



## Listing of: MONITOR.PAS

```

1  PROGRAM MONITOR ;
2  {

4      Monitor is a HALOE program designed to process HALOE
5      major frames of data sent to the IBM-XT (or compatible)
6      by the IETS HP-1000 over the HP-IB (IEEE-488 interface bus).
7      Monitor will convert the raw counts to engineering units
8      and perform limit checking and color coding of the data
9      before display on the color monitor. Monitor will also
10     archive data to disc for transfer later to an off-line
11     system for further processing and evaluation.

13     THIS PROGRAM WILL SET UP AN INTERRUPT VECTOR TO ITSELF,
14     AND LOCK ITSELF IN MEMORY TO BE CALLED BY FORTH LATER
15     USING AN INTERRUPT 48 (HEX) }

17  type
18      ivdt = record           { variable definition data }
19          leng : byte ;
20          loc : integer ;
21          bitpos, equatnum : byte ;
22          SCRPOS : INTEGER ; { SCREEN POSITION }
23          IDNAM : STRING[8] ;
24      end;
25      icoef = record         { coefficients for conversion equations }
26          slope, offset : real ;
27      end;
28      regs = record
29          AX,BX,CX,DX,BP,SI,DI,DS,ES,FLAGS : INTEGER ;
30      end;

32  var
33      REGSET : REGS ;
34      CSEGM,OFFS : INTEGER ;
35      ID1,ID2 : INTEGER ;
36      ANSWER : CHAR ;
37      VDTfileNAM : STRING[15] ;
38      vdt : ivdt ;
39      vtble : array[1..200] of ivdt ;
40      VDTFILE : FILE OF IVDT ;
41      coefFILEnam : string[15] ;
42      coef : icoef ;
43      coefTBLE : ARRAY[1..50] OF ICOEF ;
44      COEFFfile : file of icoef ;
45      WORDNUM : INTEGER ;
46      BYTEDISP : INTEGER ;
47      BITDISP : BYTE ;

50  const datseg : ARRAY[0..1] OF integer = (0,0) ;

```

## Listing of: MONITOR.FAS

```

51      STSEG : INTEGER = 0 ;
52      EXSEG : INTEGER = 0 ;
53      STPT : INTEGER = 0 ;
54      oldstseg : integer = 0 ;
55      oldstpt : integer = 0 ;
56      base : integer = $200 ;
57      HEXDIG : ARRAY[0..15] OF CHAR = '0123456789ABCDEF' ;
58  var
59      SCRNMODE : ARRAY[0..15] OF BYTE ; {DISPLAY PARAMETERS FOR GRAPHICS}
60      dataseg : ARRAY[0..1] OF integer absolute datseg ;
61      STACKSEG : INTEGER ABSOLUTE STSEG ;
62      STACKPT : INTEGER ABSOLUTE STPT ;
63      ESSEG : INTEGER ABSOLUTE EXSEG ;
64      ZILCH : integer ;
65      INTVEC : ARRAY[0..1] OF INTEGER ABSOLUTE $0000:$0120 ;
66      basearray : array[0..15] of byte absolute $0000:$0200 ;
67      year,day: string[5] ;
68      hours,minutes,seconds : string[3] ;
69      DELTA,START,STOP: REAL ;
70      sorc : string[80] ;
71  type
72      ABC = STRING[80] ;
73      cmd = string[127] ;
74      vax = string[255] ;
75      flg = integer ;
76      bad = integer ;
77      INTS = ARRAY[0..302] OF INTEGER ;
78      ANTS = ARRAY[0..604] OF BYTE ;
79      INTEGBUFF = ARRAY[0..4000] OF INTEGER ;
80      BYTEBUFF = ARRAY[0..8000] OF BYTE ;
81      HEXVAL = STRING[4] ;
82
83  var
84      COMM : INTEGER ; { HOLDS COMMAND VALUE FROM ODD OR EVEN COMMAND WORD }
85      INDEX : INTEGER ; { COMM IS USED TO CALCULATE INDEX OF COMMAND IN TBLE }
86      port21 : byte ; { 8259 interrupt mask register }
87      txt : text ;
88      txtfile : string[10] ;
89      att : integer ;
90      I,J,ind:INTEGER ;
91      COUNT : INTEGER ;
92      syscon:cmd ;
93      f:flg ;
94      b:bad ;
95      needmoredata : boolean ;
96      STATUS : INTS ;
97      STAT : VAX ABSOLUTE STATUS ;
98      numsaddr : INTS ;
99      NUMSAD : VAX ABSOLUTE NUMSADDR ;
100     c:cmd ;
101     IANS: CHAR ;

```

## Listing of: MONITOR.PAS

```

102     NUMS: INTS ;
103     BYTES : ANTS ABSOLUTE NUMS ;
104     V : VAX ABSOLUTE NUMS ;
105     TEMP : BYTE ;
106     FRAME : INTEGbuff ABSOLUTE $B800:$0000;
107     BFRAME : BYTEbuff ABSOLUTE $B800:$0000;
108     TIMER : BYTE ABSOLUTE $0040:$006C ;
109     mask,mask2,num1,num2,shift: integer ; { used by bits function }
110     LINENUM,CHARNUM : INTEGER ;
111     SCRINT : ARRAY10..11 OF INTEGER ABSOLUTE $0000:$0014 ;
112     STORINT : ARRAY10..11 OF INTEGER ;
113     STATPR : BYTE ABSOLUTE $0050:$0000 ;

116     PROCEDURE SETINTVEC(SEGM,OFFS:INTEGER) ;
117     { set up interrupt vector number $48 (hex) to point to
118       the main subroutine }
119     var ah,al : byte ;
120     BEGIN
121         WITH REGSET DO
122             BEGIN
123                 DS := SEGM ;
124                 DX := OFFS ;
125                 ah := $25 ;
126                 AX :=( ah shl 8) or $48 ;
127                 INTR($21,REGSET);
128             END;
129     END;

132     FUNCTION HEX(VAL:INTEGER): HEXVAL ;
133     { convert val into a hex string }
134     BEGIN
135         HEX := HEXDIG[VAL SHR 12] +
136               HEXDIG[(VAL SHR 8) AND 15] +
137               HEXDIG[(VAL SHR 4) AND 15] +
138               HEXDIG[VAL AND 15] ;
139     END;

143     FUNCTION BITS(NUMS:ints;IND:INTEGER;BITPOS,LENGTH:BYTE): INTEGER ;
144     { extract length bits from bitpos of nums[ind] }
145     BEGIN
146         BITPOS := 16 - BITPOS ;
147         NUM1 := NUMS[IND];
148         NUM2 := NUMS[IND+1] ;
149         SHIFT := BITPOS - LENGTH ;
150         IF SHIFT < 0 THEN
151             BEGIN
152                 MASK := ($FFFF SHR (16 - BITPOS)) ;

```

## Listing of: MONITOR.PAS

```

153     MASK2 := $FFFF SHR (16+SHIFT) ;
154     BITS := ((NUM1 AND MASK) SHL -SHIFT) OR
155             ((NUM2) SHR (16 + SHIFT)) AND MASK2 ;
156     END
157     ELSE
158     IF SHIFT = 0 THEN
159     BEGIN
160         MASK := $FFFF SHR (16 - LENGTH) ;
161         BITS := MASK AND NUM1 ;
162     END
163     ELSE
164     BEGIN
165         MASK := $FFFF SHR (16 - LENGTH) ;
166         BITS := (NUM1 SHR SHIFT) AND MASK ;
167     END;
168 END;

```

```

171 procedure SCRDUMP(var i,j: integer) ;
172 TYPE CHARBUFF = ARRAY[0..8000] OF CHAR ;
173 VAR CFRAME: CHARBUFF ABSOLUTE $B800:$0000;
174 PRFRAME: ARRAY[0..4000] OF CHAR ;
175 K,l: INTEGER ;
176 begin
177 IF (I+J = 0) THEN
178 BEGIN
179 FOR K := 0 TO 3999 DO
180 BEGIN
181 PRFRAME[K] := CFRAME[K*2];
182 END;
183 END;
184 for l:= 0 to 4 do
185 begin
186     if (j<79) then
187     begin
188         WRITE(LST,PRFRAME[I*80 +j]) ;
189     end
190     else
191     begin
192         writeln(lst,PRFRAME[I*80+j]);
193     end;
194     j:= j+1;
195 end;
196 if (j>79) then
197 begin
198     j:=0;
199     i := i+1;
200     if (i>48) then
201     begin
202         i := 0;
203         statpr :=0;

```

## Listing of: MONITOR.PAS

```

204     end;
205     end;
206     end;

210     FUNCTION STACK : INTEGER ; EXTERNAL 'STACK.COM' ;
211     { STACK RETURNS VALUE OF STACK POINTER }

213     FUNCTION ESEGM : INTEGER ; EXTERNAL 'ESEG.COM' ;
214     { RETURNS VALUE OF ES ..EXTRA SEGMENT REGISTER }

216     procedure IE488 ( VAR c:cmd;
217                      VAR v:vax;
218                      VAR f:flg;
219                      VAR b:bad ); external 'IE488.COM';

224     PROCEDURE S5080(var i :byte); EXTERNAL 'CONO.COM';
225     { S5080 PUTS THE CONOGRAPHICS SYSTEM IN THE DESIRED MODE:
226       At program start, it puts the screen in 50 row,80 column mode.
227       At termination, it returns the screen to 25 X 80 . }

229     PROCEDURE PUTOUT(VAR SORC:ABC;VAR FRAME:INTEGER;ATTR:INTEGER);
230     EXTERNAL 'PUTOUT.COM';
231     { PUTOUT places a string and its color attributes
232       in the screen memory area }

234     FUNCTION PRSTAT:INTEGER; EXTERNAL 'PRSTAT.COM';
235     { PRSTAT responds to the shift-PrtSC keys by setting a flag.
236       The program will then dump the screen to the printer
237       50 rows by 80 columns }

239     FUNCTION XYPOS(ROW,COL:INTEGER ):INTEGER ;
240     BEGIN
241         XYPOS := ROW * 80 + COL;
242     END;

246     procedure NEWSCREEN ;
247     { set up conographics screen mode with 80 columns and 50 rows }
248     BEGIN
249         SCRNMODE[0] := $71;
250         SCRNMODE[1] := $50;
251         SCRNMODE[2] := $5A;
252         SCRNMODE[3] := $0F;
253         SCRNMODE[4] := $1B;
254         SCRNMODE[5] := 6;

```

## Listing of: MONITOR.PAS

```

255     SCRNM0DE[6] := $19;
256     SCRNM0DE[7] := $1A;
257     SCRNM0DE[8] := 3;
258     SCRNM0DE[9] := 7;
259     SCRNM0DE[10] := $20 ;
260     SCRNM0DE[11] := $20 ;
261     SCRNM0DE[12] := 0;
262     SCRNM0DE[13] := 0;
263     SCRNM0DE[14] := 0;
264     SCRNM0DE[15] := 0;
265     S5080(SCRNM0DE[0]);

267  END;

269  PROCEDURE OLDScreen ;
270  { restore old screen mode }
271  VAR LOC : INTEGER ;
272  BEGIN
273      FOR LOC := 0 TO 3999 DO
274          FRAME[LOC] := $FOO ;

276          SCRNM0DE[4] := $1F ;
277          SCRNM0DE[7] := $1C ;
278          SCRNM0DE[8] := 2;
279          SCRNM0DE[10] := 6;
280          SCRNM0DE[11] := 7;
281          S5080(SCRNM0DE[0]);
282  END;

284  PROCEDURE DISPLAYACRO ;
285  { display background for limit check screen }
286  VAR I: INTEGER;
287  BEGIN
288      txtfile := 'HALOE.SCR' ;
289      assign(txt,txtfile);
290      reset(txt);
291      att := 15 ;
292      i := 0 ;
293      while not eof(txt) do
294      begin
295          readln(txt,sorc);
296          sorc := sorc + ' ';
297          putout(sorc,frame[i],att);
298          i := i + 80 ;
299      end;
300      close (txt);

302  END;

```

## Listing of: MONITOR.PAS

```

306  { -----PROCEDURES & FUNCTIONS-----}

308  PROCEDURE OUTPUT(VAR SORC: ABC; VAR FRAME: INTEGER; ATTR: INTEGER);
309  VAR BLANKS : ABC ;
310  BEGIN
311  {      BLANKS := '          ' ; 10 BLANKS }
312  {      PUTOUT(BLANKS,FRAME,WHITE);    }
313  PUTOUT(SORC,FRAME,ATTR);
314  END;

316  function bcd2dec(x:integer):integer;
317  { convert bcd value x into decimal value }
318  begin
319      bcd2dec :=(x div 16 )*10 + (x mod 16) ;
320  end;

323  FUNCTION TIME: REAL ;
324  CONST TIMEBASE = 893 ;
325  VAR TSEC,HUNDSEC,SEX,MENS: INTEGER ;
326  BEGIN
327      PORT[TIMEBASE] := 0 ;    { SELECT THOUSANDTHS OF SECONDS REGISTER }
328      TSEC := BCD2DEC(PORT[TIMEBASE+2]);
329      PORT[TIMEBASE] := 1 ;    { SELECT HUNDREDTHS AND TENTHS REGISTER }
330      HUNDSEC := BCD2DEC(PORT[TIMEBASE+2]);
331      PORT[TIMEBASE] := 2 ;
332      SEX := BCD2DEC(PORT[TIMEBASE+2]) ;
333      PORT[TIMEBASE] := 3 ;
334      MENS := BCD2DEC(PORT[TIMEBASE+2]) ;

336      TIME := TSEC/1000. + HUNDSEC/100. + SEX + MENS*60.0;
337  END;

339  PROCEDURE DISPLAY(ITEM,NDEC,XPOS,YPOS,COLR: INTEGER ) ;
340  BEGIN
341      STR(ITEM:NDEC,SORC);
342      PUTOUT(SORC,FRAME[XYPOS(XPOS,YPOS)],COLR);

344  END;

346  PROCEDURE DISPLAYDATA ;
347  VAR VALU,K : INTEGER ;
348      xvalu : real ;
349      VDT1,VDT2 : IVDT ;
350      KDV,XV,XDV,BBI,BBV : REAL ;

352  CONST COLON : CHAR = ':' ;
353  LABEL THEexit ;
354  BEGIN

356      str(nums[10]:5,year);

```

## Listing of: MONITOR.PAS

```

357      str(nums[9]:5,day);
358      str(nums[8]:3,hours);
359      str(nums[7]:2,minutes);
360      str(nums[6]:2,seconds);
361      sorc := year + day + hours + COLON
362            + minutes + COLON + seconds ;
363      putout(sorc,frame[xypos(0,57)],yellow);
364      FOR I:= 1 TO 10 DO
365      BEGIN
366          VDT := VTBLE[I] ;
367          IF VDT.SCRPOS > 0 THEN
368          BEGIN
369              K := VDT.LOC -1 ;
370              VALU := BITS(NUMS,K,VDT.BITPOS,VDT.LENG);
371              J := VDT.EQUATNUM ;
372              IF J > 128 THEN J := J -256 ;
373              IF J > 0 THEN
374              BEGIN
375                  if ( J < 51) and (j <> 2) then
376                  begin
377                      COEF := COEFTBLE[J] ;
378                      xvalu := valu*(COEF.SLOPE) + COEF.OFFSET ;
379                      str(xvalu:6:3,sorc);
380                  end
381                  else
382                  IF (I=94) OR (I=95) THEN
383                  BEGIN { EVEN OR ODD COMMAND }
384                      SORC := HEX(VALU) + ' ' ;
385                      PUTOUT(SORC,FRAME[VDT.SCRPOS],GREEN);
386                      COMM := VALU SHR 12 ; { GET COMMAND NUMBER }
387                      VALU := VALU AND 4095 ;
388                      CASE COMM OF
389                      1,3,5,7,9,11,13,15: INDEX := COMM div 2 + 110 ;
390                      0,2,4,6,8,10,12,14: INDEX := COMM div 2 + 100 ;
391                      END;
392
393                      VDT := VTBLE[INDEX] ; { SELECT TABLE ENTRY FOR
394                                              THIS COMMAND}
395                      sorc := hex(valu);
396                  END
397                  else
398                      STR(VALU:6,SORC);
399              END
400          ELSE
401          BEGIN
402              { SPECIAL PROCESSING.. EQUIVALENT OF ISPCL IN HP SOFTWARE}
403              J := ABS(J) -25 ;
404              IF (J<0) OR (J>6) THEN GOTO THEexit ;
405              CASE J OF
406              1,2,3,4 :
407              BEGIN

```



## Listing of: MONITOR.PAS

```

408      ID2 := 2*J ;
409      ID1 := ID2 -1 ;
410      VDT1 := VTBLE[ID1] ;
411      VDT2 := VTBLE[ID2] ;
412      K := VDT1.LOC -1 ;
413      XV := BITS(NUMS,K,VDT1.BITPOS,VDT1.LENG);
414      COEF := COEFTBLE[VDT1.EQUATNUM] ;
415      XV := XV*COEF.SLOPE + COEF.OFFSET ;
416      K := VDT2.LOC -1 ;
417      XDV := BITS(NUMS,K,VDT2.BITPOS,VDT2.LENG);
418      COEF := COEFTBLE[VDT2.EQUATNUM] ;
419      XDV := XDV*COEF.SLOPE + COEF.OFFSET ;
420      KDV := 58.0;
421      IF J=4 THEN
422          BEGIN
423              XDV := XDV + 4.639 ;
424              KDV := 29.0 ;
425          END;
426      XVALU := XDV/KDV*1.E6 ;
427      END;
428      5,6:
429      BEGIN
430          VDT1 := VTBLE[21] ; { BBI }
431          VDT2 := VTBLE[48] ; { BBV }
432          K:= VDT1.LOC -1 ;
433          BBI :=BITS(NUMS,K,VDT1.BITPOS,VDT1.LENG);
434          COEF := COEFTBLE[VDT1.EQUATNUM];
435          BBI := BBI*COEF.SLOPE + COEF.OFFSET ;
436          K := VDT2.LOC -1 ;
437          BBV := BITS(NUMS,K,VDT2.BITPOS,VDT2.LENG);
438          COEF := COEFTBLE[8] ;
439          bbv := bbv*coef.slope + coef.offset ;
440          XVALU := BBV - BBI*0.5 ;
441          IF J=6 THEN XVALU := XVALU/BBI ;
442      END ;
443      END; { OF CASE }
444      STR(XVALU:10:4,SORC);
445      END;
446      PUTOUT(SORC,FRAME[VDT.SCRPOS],GREEN);
447      TheExit:      END;
448      END;
449      END;

453      procedure main ;
454      begin
455          port21 := port[#21] ;
456          port[#21] := port21 or 1 ;
457          numsaddr[0] := seg(nums[0]) ;
458          numsaddr[1] := ofs(nums[0]) ;

```

## Listing of: MONITOR.PAS

```

459 { CHECK FOR SCREEN DUMP }
460 { IF STATPR = 1 THEN
461 REPEAT
462     SCRUMP(LINENUM,CHARNUM);
463 UNTIL STATPR = 0 ;}
464     STR(TIMER:4,SORC);
465     SORC := 'TIMER = ' + SORC ;
466     PUTOUT(SORC,FRAME[1220],WHITE);

469     if needmoredata then
470     begin
471         C:= 'ENTER [WD,0,301]' ; { set up for DMA transfer of 604 bytes }
472         IE488(C,numsad,F,B); { input 302 words of data input v array }
473         needmoredata := FALSE ;
474         { START := TIME ;}
475     end
476     else
477     begin
478         { COUNT := COUNT + 1 ;}
479         C:= 'REQUEST' ;
480         STATUS[0] := 0 ;
481         IE488(C,STAT,F,B);
482         if ((STATUS[0] AND $200) = 0) then
483         begin
484             { STOP := TIME ;}
485             { DELTA := STOP - START ;}
486             { WRITELN(' ITERATIONS = ', COUNT:5, ' TIME = ',DELTA:10:5);}
487             { COUNT := 0 ;}
488             FOR I := 1 TO 302 DO
489             BEGIN
490                 J := 2*I ;
491                 TEMP := BYTES[J] ;
492                 BYTES[J] := BYTES[J+1] ;
493                 BYTES[J+1] := TEMP ;
494             end;
495             displaydata ; needmoredata := true ;
496         end;
497     end;
498 port[$21] := port21 ; { restore interrupt mask for 8259 }
499 end;

502 procedure INTieee;
503 begin
504     inline( $FB/ { STI ENABLE INTERRUPTS }
505             $50/ { PUSH AX }
506             $53/ { PUSH BX }
507             $51/ { PUSH CX }
508             $52/ { PUSH DX }
509             $56/ { PUSH SI }

```

## Listing of: MONITOR.PAS

```

510          $57/          { PUSH DI }
511          $1e/          { PUSH DS }
512          $06/          { PUSH ES }
513          $55           { PUSH BP }
514          );
515  INLINE($2E/$C5/$3E/DATSEG); { SET DS REG TO DATA SEG }
516  inline(
517          $1e/          { push ds }
518          $07 ) ;        { pop es } { turbo ds & es are same }
519  inline($2e/$89/$26/oldstpt); { save old stack pointer }
520  INLINE($2E/$8B/$26/STPT); { SET STACK POINTER }
521  inline($2e/$8c/$16/oldstseg); { save old stack seg }
522  INLINE($2E/$8E/$16/STSEG); { SET STACK SEGMENT REG }
523  MAIN ; { CALL MAIN PROCEDURE }
524  inline($2e/$8b/$26/oldstpt); { restorr old stack pointer }
525  inline($2e/$8e/$16/oldstseg); { restore old stack segment }

527  inline( $5d/          { POP BP }
528          $07/          { POP ES }
529          $1f/          { POP DS }
530          $5f/          { POP DI }
531          $5e/          { POP SI }
532          $5a/          { POP DX }
533          $59/          { POP CX }
534          $5b/          { POP BX }
535          $58/          { POP AX }

537          $cf          { IRET }
538          ); { RETURN TO 4TH }

540  end;

545  { -----PAS4TH MAIN PROGRAM----- }

548  BEGIN
549  COUNT := 0 ;
550  newscreen ;
551  displayacro ;
552  needmoredata := true ;
553  f := 1;
554  b := 0;
555  STORINT[0] := SCRINT[0] ; {SAVE PRINT SCREEN VECTOR }
556  STORINT[1] := SCRINT[1] ;
557  SCRINT[0] := OFS(PRSTAT) ;
558  SCRINT[1] := CSEG ;
559  STATPR := 0 ;
560  LINENUM := 0 ;

```

Listing of: MONITOR.PAS

```

561  CHARNUM := 0 ;

563  { get coefficient file name }

565  writeln(' enter coefficient file name (usually coef.dat)');
566  readln(coefFileName);
567  {coefFileName := 'coef.dat' ;}
568  assign(coefFILE,COEFfileNAM);
569  RESET(COEFFILE);

571  { read in coefficients }
572  FOR I:= 1 TO 50 DO
573  READ(COEFFILE,COEFTBLE[I]);
574  CLOSE(COEFFILE) ;

576  { get variable definition table file name }
577  WRITELN(' ENTER VARIABLE DEFINITION FILE NAME (USUALLY VARDEF.DAT)');
578  READLN(VDTFILENAM);
579  ASSIGN(VDTFILE,VDTFILENAM);
580  RESET(VDTFILE);
581  { read in variable definition table }
582  FOR I:= 1 TO 200 DO
583  READ(VDTFILE,VTBLE[I]);      { READ IN THE VARIABLE DEFINITION TABLE }
584  CLOSE(VDTFILE) ;

586  { set up ieee-488 bus. my address = 3 (MAD=3)
587  computer in charge= 1, number of ieee-488 cards = 1,
588  base address for ieee card = 200 hex }
589  syscon := 'SYSCON MAD=3, CIC=1, NOB=1, BAO=&H200';
590  v := ' ';
591  { send initialization command contained in string syscon }
592  IE488(syscon,v,f,b);
593  f :=2;
594  b :=0;
595  C:= 'PASCTL 0' ;
596  { writeln('PASSING CONTROL TO HP'); }
597  { need to send control to HP-1000 }
598  IE488(c,v,f,b);
599  F:= 0;
600  B:= 0 ;
601  C := 'TIMEOUT' ;
602  V := chr(1) ;
603  { set up for infinite time out value }
604  IE488(C,V,F,B);

606  ESSEG := ESEGM ;
607  dataseg[0] := DSEG ;
608  DATASEG[1] := DSEG ;
609  WRITELN(' ESEG & DESG =',ESSEG:6,DATASEG[0]:6);
610  STACKSEG := SSEG ;
611  STACKPT := STACK ;

```

Listing of: MONITOR.PAS

```
613      csegm := cseg ;
614      offs := ofs(INTieee)+ 7 ; { THE + 7 SKIPS OVER TURBO PROCEDURE CODE }
615      SETINTVEC(CSEGM,OFFS);
616      writeln('      PAS4TH CS,OFFS :',intvec[0]:6,intvec[1]:6);
617      { writeln('      Datseg =',datseg[0]:6,datseg[1]:6); }
618      { port[$20B] := 1 ; }

620 { interrupt type 27 hex allows a program to terminate while locking
621   itself in memory. This main program is never re-entered, but interrupt
622   type 48 hex will cause the main procedure to be called which in turn
623   utilizes the rest of this program code }
624   intr($27,zilch);
625   END.
```

# Standard Bibliographic Page

1. Report No. NASA CR-178339		2. Government Accession No.		3. Recipient's Catalog No.	
4. Title and Subtitle  HALOE TEST AND EVALUATION SOFTWARE				5. Report Date June 1987	
				6. Performing Organization Code	
7. Author(s) W. Edmonds S. Natarajan				8. Performing Organization Report No.	
				10. Work Unit No. 678-12-04-09	
9. Performing Organization Name and Address ST Systems Corporation (STX) 28 Research Drive Hampton, VA 23666				11. Contract or Grant No. NAS1-18022	
				13. Type of Report and Period Covered Contractor Report	
12. Sponsoring Agency Name and Address NASA Langley Research Center Hampton, VA 23665				14. Sponsoring Agency Code	
15. Supplementary Notes  Langley Technical Monitor: R. L. Baker					
16. Abstract  Computer programming, system development and analysis efforts during this contract were carried out in support of the Halogen Occultation Experiment (HALOE) at NASA/Langley. Support in the major areas of data acquisition and monitoring, data reduction and system development are described along with a brief explanation of the HALOE project. Documented listings of major software are located in the appendix.					
17. Key Words (Suggested by Authors(s)) HALOE, UARS, ozone depletion, software development, system development, data acquisition data monitoring, Forth, Pascal				18. Distribution Statement  Unclassified - Unlimited  Subject Category 61	
19. Security Classif.(of this report) Unclassified		20. Security Classif.(of this page) Unclassified		21. No. of Pages 142	
				22. Price	

For sale by the National Technical Information Service, Springfield, Virginia 22161